Modelling the BMA-Process with MATLAB

Case Study

Ramona Achermann, D-CHAB, 6th semester, *ramonaa@student.ethz.ch*
Tim Forster, D-CHAB, 6th semester, *forstert@student.ethz.ch*
Dany Liu, D-CHAB, 6th semester, *liud@student.ethz.ch*
Shant Dakessian, D-CHAB, 6th semester, *dshant@student.ethz.ch*

Abstract: A chemical plant for the production of HCN was modelled entirely in MATLAB. Four units were considered: the reactor, an ammonia absorber, an HCN absorber, and a distillation unit. A production target of 10 kt/y was set. The modelling was done for both ideal and non-ideal thermodynamic models. The yield from the reactor was modelled for different gas pressures and methane to ammonia ratios, where an optimum was reached with 5.6 mol/s NH_3 inlet feed for the non-ideal case and 12.8 mol/s for the ideal case. In the NH_3 absorber 10 mol-% sulfuric acid was used as an absorption medium. The HCN was then absorbed using water, and distilled to 99.5% purity. The bottoms of the distillation was recycled as wash water for the absorption. A cost analysis was subsequently carried out. The total cost over the 10 year depreciation period were 65.8 mio. USD and the total revenue due to H_2 and HCN sale was 178 mio. USD, which lead to an overall profit of 112 mio. USD.

Ramona Achermann Tim Forster Dany Liu Shant Dakessian

Zurich, June 3, 2016

Contents

1 Introduction

Hydrogen cyanide is an important building block for several industrially important substances such as cyanuric chloride, adiponitrile and methyl methacrylate. These are used in a wide range of products including nylon, plastics (PMMA) and pesticides. In today's industry the two most common processes for the production of hydrogen cyanide are the Andrussow process, developed by Dr. L. Andrussow in 1955, and the Degussa process, also known as the BMA process which was discovered by the German company Degussa a short while later. The abbreviation stands for "Blausäure aus Methan und Ammoniak". This case study has the objective of evaluating the BMA process in a plant with a production volume of 10'000 tons/year.

$$NH_3 + CH_4 \longrightarrow HCN + 3\,H_2$$
$$2\,NH_3 \longrightarrow N_2 + 3\,H_2$$

The Degussa process (BMA) uses methane (CH_4) and ammonia (NH_3) to produce hydrogen cyanide (HCN), and hydrogen gas (H_2) as a byproduct. The main difficulties in this reaction is the relatively high temperature of at least 1000 °C at which it must be conducted, which could lead to coke formation as well as the separation and purification of the product from the unreacted starting materials, and the byproducts from the second reaction. The high temperatures increase the overall yield as the desired reaction is more endothermic than the decomposition of ammonia, and is therefore favored at high temperatures. This process is used less commonly as the requirements are much higher; the reactor and entire plant is more expensive and complex. The benefit of the special ceramic tubes (Al_2O_3) coated with the platinum catalyst gives a 90 % selectivity of ammonia to hydrogen cyanide [1]. Additionally, the hydrogen gas produced in three equivalents is pure and can be used in other processes or to produce the heat for this process.
A flowsheet of the plant which was simulated in this case study is shown in Figure 1. The purified NH_3 and CH_4 are preheated in a heat exchanger from room temperature to 700 K and is then sent to the reactor. The energy which is needed for the endothermic reaction to run will be provided by burning natural gas. Before the outlet stream is sent to a purification unit, it is sent again through the heat exchanger from before. In this case, some of the hot outlet stream can be used to heat the reactor feed stream and energy as well as costs can be saved. NH_3 will be absorbed in the first absorber unit using an aqueous sulfuric acid solution (H_2SO_4). In a second absorber unit, the product HCN will be separated from the rest by using water. Lastly, the product will be obtained by removing the water through distillation. The last two units are coupled with each other. The water stream which is almost free of HCN is sent back and used as the inlet water stream for the HCN absorption.

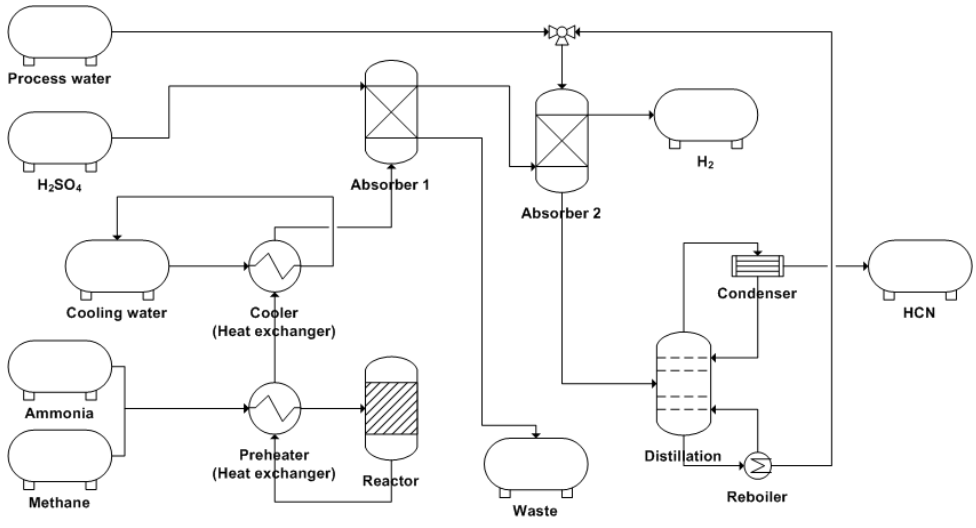

Figure 1: Process flowsheet of the simulated plant. The process can be divided in *reaction*, *separation* and *purification*.

2 Methods and Calculations

2.1 Calculations Reactor

2.1.1 Optimization of the yield

One of the first issue to be considered was the pressure in the reaction. Analysis of the Navier-Stokes-equation[2][1] one recognizes that a constant pressure will lead to a velocity, which is constant.

$$\frac{\partial u_x}{\partial t} + \left[u_x \frac{\partial u_x}{\partial x} + u_y \frac{\partial u_x}{\partial y} + u_z \frac{\partial u_x}{\partial z} \right] = -\frac{1}{\rho} \frac{\partial p}{\partial x} + \nu \left[\frac{\partial^2 u_x}{\partial x^2} + \frac{\partial^2 u_y}{\partial y^2} + \frac{\partial^2 u_z}{\partial z^2} \right] + f_i \tag{1}$$

$$u_x \frac{\partial u_x}{\partial x} = -\frac{1}{\rho} \frac{\partial p}{\partial x} \tag{2}$$

Due to the fact that the reactor consists of pipes with coated walls which are relatively short, the assumption was made that the pressure remains constant. Therefore, the pressure was set to 1 atm and kept constant over the whole reactor. Nevertheless, due to an increase in the amount of moles during the reaction, the velocity must be assumed to increase (see Table 4) during the simulation. This can be seen in an increase of the flow rate of the reaction gas.

$$\mathrm{NH_3 + CH_4 \longrightarrow HCN + 3\,H_2}$$
$$\mathrm{2\,NH_3 \longrightarrow N_2 + 3\,H_2}$$

In order to optimize the yield with respect to the feed rate of NH_3 a mass and an energy balance had to be constructed. Therefore, the differential equation of the feed rate over the volume was taken for the mass balance of the PFR:

$$\frac{dF_i}{dV} = r_i \cdot a \tag{3}$$

Equation (3) was written for each of the five substances.

$$\frac{dF_{NH_3}}{dV} = -r_{HCN}a - 2r_{N_2}a \tag{4}$$

$$\frac{dF_{CH_4}}{dV} = -r_{HCN}a \tag{5}$$

$$\frac{dF_{HCN}}{dV} = r_{HCN}a \tag{6}$$

$$\frac{dF_{H_2}}{dV} = 3r_{HCN}a + 3r_{N_2}a \tag{7}$$

$$\frac{dF_{N_2}}{dV} = r_{N_2}a \tag{8}$$

The reaction rates r_i were given in the documentation by equations (9) and (10), respectively. The surface-volume ratio a was calculated through the fraction between the surface area of the cat (inner surface area) and the volume.

$$r_{HCN} = \frac{7.8 \cdot 10^{18} \exp(-\frac{1950}{T}) p_{CH_4} p_{NH_3}^{0.5}}{(1 + 0.044 \exp(\frac{2390}{T})(p_{CH_4} p_{NH_3}^{-0.5}))^4} \tag{9}$$

$$r_{N_2} = \frac{4.9 \cdot 10^{18} \exp(-\frac{2130}{T}) p_{NH_3}}{(1 + 0.044 \exp(\frac{2390}{T})(p_{CH_4} p_{NH_3}^{-0.5}))^3} \tag{10}$$

In order to make the ordinary differential equations dependent on the feed streams, the partial pressures p_i in the above reaction rate equations were rewritten as a function of the feed streams of the chemicals F_i, the total feed stream F_{tot} and the pressure p as shown in the equation below.

[1] In pipe/x direction, in two dimensions for simplicity, solid wall, small viscosity and no volume forces.

$$p_i = \frac{F_i}{F_{tot}} p \tag{11}$$

The total feed stream was calculated for each position z in the reactor.

$$F_{tot} = \sum_{i=1}^{5} F_i \tag{12}$$

With the above written equations the ordinary differential equations can be solved in MATLAB using the solver *ode*15*s* and the feed streams can be plotted against the reactor length by dividing the volumes by the cross-sectional area.
In order to get a temperature profile over the reactor length an energy balance was derived:

$$\frac{dT}{dV} = \frac{Ua(T_a - T) - \sum r_i \Delta_r H}{\sum F_i^{in} c_p} \tag{13}$$

U is the heat transfer coefficient of the pipe wall (later the overall heat transfer coefficient), T_a the outer temperature of the reactor pipe (given at 1600 K), T the variable temperature, $\Delta_r H$ the reaction enthalpies [1] and C_p the heat capacities. The latter are dependent on the temperature, which changes over the length of the reactor. Therefore, they had to be calculated for each position z in the reactor using the Shomate equation[3]:

$$C_{p,i}(T) = A_i + B_i \cdot t + C_i \cdot t^2 + D_i \cdot t^3 + \frac{E_i}{t^2} \tag{14}$$

where $t = T/1000$ and T represents the temperature. The coefficients A_i, B_i, C_i, D_i and E_i are different for each component i and are tabulated in literature [3].

2.1.2 Study of Parameter Dependence on Feed Rate and Influence of Other Thermodynamic Models of the Gas Phase

In order to get an outer temperature profile the energy balance over the reactor (equation (13)) was used again, but this time the radial influence was considered using a different overall heat transfer coefficient U than before:

$$U = \frac{1}{\frac{1}{\alpha_{in}} + \frac{1}{\alpha_{wall}} + \frac{1}{\alpha_{out}}} \tag{15}$$

The dynamic viscosity μ as well as the thermal conductivity α depend on temperature, which is changing over the whole reactor. Therefore, μ and α are changing as well and had to be calculated for each location in the reactor pipe and for each temperature in the ODE. First literature values were found for each substance at different temperatures [4]. Then, these values were extrapolated with a linear function for the thermal conductivity [5] and with a quadratic function for the dynamic viscosity for temperatures above 700 K by using the equations (16)-(20). The temperature-dependent equations for λ were found for each substance by linear regression and the overall temperature dependence of the thermal conductivity of the reaction mixture was found assuming an ideal mixture (equation (21)).

$$\lambda_{NH_3} = 1.41 \cdot 10^{-4} T - 0.0186 \tag{16}$$

$$\lambda_{CH_4} = 1.56 \cdot 10^{-4} T - 0.0110 \tag{17}$$

$$\lambda_{HCN} = 1.27 \cdot 10^{-4} T - 0.0084 \tag{18}$$

$$\lambda_{H_2} = 5.41 \cdot 10^{-4} T + 0.0192 \tag{19}$$

$$\lambda_{N_2} = 6.75 \cdot 10^{-4} T + 0.0046 \tag{20}$$

$$\lambda_{tot} = \sum \frac{F_i}{F_{tot}} \cdot \lambda_i \tag{21}$$

The temperature dependent equations for μ were found for each substance by quadratic regression and the overall temperature dependence of the dynamic viscosity of the reaction mixture was found assuming an ideal mixture (equation (27)).

$$\mu_{NH_3} = -8.28 \cdot 10^{-13} T^2 + 3.89 \cdot 10^{-8} T - 1.42 \cdot 10^{-6} \tag{22}$$
$$\mu_{CH_4} = -1.12 \cdot 10^{-11} T^2 + 3.77 \cdot 10^{-8} T - 9.21 \cdot 10^{-7} \tag{23}$$
$$\mu_{HCN} = -1.81 \cdot 10^{-11} T^2 + 5.96 \cdot 10^{-8} T - 1.26 \cdot 10^{-6} \tag{24}$$
$$\mu_{H_2} = -3.68 \cdot 10^{-12} T^2 + 2.07 \cdot 10^{-8} T + 3.07 \cdot 10^{-6} \tag{25}$$
$$\mu_{N_2} = -1.60 \cdot 10^{-11} T^2 + 5.33 \cdot 10^{-8} T + 3.35 \cdot 10^{-6} \tag{26}$$

$$\mu_{tot} = \sum \frac{F_i}{F_{tot}} \cdot \mu_i \tag{27}$$

For each of the resistances another heat transfer coefficient α was calculated using the dimensionless numbers Prantl, Reynolds and Nusselt, respectively. The Prandtl number was calculated through equation (28).

$$Pr = \frac{C_p \cdot \mu}{\lambda} \tag{28}$$

To calculate the Pr-number, the kinematic viscosity ν was needed. This property was determined by using the definition of ν (29) and the ideal gas law (30).

$$\nu_{tot} = \frac{\mu_{tot}}{\rho} \tag{29}$$
$$\rho = \frac{pM}{RT} \tag{30}$$

The Prandtl number changed over the reactor length but varies around the value of $Pr = 0.8$ which is consistent with values found in the literature [6]. Then, the Reynolds number was calculated using the following expression:

$$Re = \frac{v \cdot d \cdot \rho}{\mu} \tag{31}$$

The velocity v was calculated using the ideal gas law (equation (32)), the density ρ through equation (30) and for the characteristic length d the diameter of the pipe was taken.

$$v = \frac{F \cdot R \cdot T}{p \cdot A} \tag{32}$$

Due to the fact that the Reynolds numer is higher than 2'300 (Table 3), which is the value where the laminar flow changes to a turbulent one in a pipe, the following relation can be used for the Nusselt number[6]. It is specific for a smooth pipe with longitudinal flow and considers the property variation of the fluid due to wide temperature ranges:

$$Nu = 0.027 \cdot Re^{0.80} \cdot Pr^{1/3} \cdot (\mu/\mu_S)^{0.14} \tag{33}$$

Lastly, α can be calculated through the Nusselt number.

$$\alpha_{in} = \frac{Nu \cdot \lambda_{Fluid}}{d} \tag{34}$$

This newly calculated overall[2] heat transfer coefficient U (equation (15)) was used in the energy balance of the reactor (equation (13)) and new temperature and flow profiles were obtained, which were again optimized (Additionally shown in Figure 17, 15 and 16).
In a last step, the behaviour of each substance in the gas mixture was considered to be real instead of ideal. To account for non-ideality in the gas phases, the fugacity was used instead of the total pressure in all equations above,

$$f = P \cdot \phi \tag{35}$$

By using the Peng-Robinson equation of state[7], the fugacity coefficient is calculated by,

$$\ln \phi = Z - 1 - \frac{A}{2\sqrt{2}B} \ln\left(\frac{Z + B(1 + \sqrt{2})}{Z + B(1 - \sqrt{2})}\right) - \ln(Z - B). \tag{36}$$

So the compressibility factor Z was determined by solving,

$$Z^3 - (1 - B)Z^2 + (A - 2B - 3B^2)Z - (AB - B^2 - B^3) = 0 \tag{37}$$

and the largest Z was used for the gas. The parameters A and B are given by,

$$A = \frac{akp}{R^2T^2} \qquad B = \frac{bp}{RT} \tag{38}$$

$$a = \frac{0.457235R^2T_c^2}{p_c} \qquad b = \frac{0.077796RT_c}{p_c} \tag{39}$$

$$s_1 = 0.37464 \quad s_2 = 1.54226 \quad s_3 = -0.26992 \tag{40}$$

$$k = (1 + S(1 - \sqrt{T_R}))^2 \tag{41}$$

$$S = s_1 + s_2\omega + s_3\omega^2 \tag{42}$$

To get the critical pressure and critical temperature (shown in Table 1 for each species) of the mixture, the combination rules were used. For the parameter a the geometric average,

$$a_{mix} = \left(\sum_i x_i \sqrt{a_i}\right)^2 \qquad \text{with} \qquad x_i = \frac{F_i}{F_{tot}}, \tag{43}$$

for the parameter b the arithmetic average,

$$b_{mix} = \sum_i x_i b_i \qquad \text{with} \qquad x_i = \frac{F_i}{F_{tot}}, \tag{44}$$

and for the parameter ω (acentric factor), as well the arithmetic average,

$$\omega_{mix} = \sum_i x_i \omega_i \qquad \text{with} \qquad x_i = \frac{F_i}{F_{tot}}, \tag{45}$$

were used to calculate the parameters of the gas mixture. Additionally, the real flow velocity u_{real} of the gas in the pipes was calculated by using the following expressions. The Peng-Robinson equation was written explicitly for the pressure[7],

$$P = \frac{RT}{\tilde{v}_m - b_{mix}} - \frac{a_{mix}}{\tilde{v}_m^2 + 2\tilde{v}_m b_{mix} - b_{mix}^2}. \tag{46}$$

By solving this equation for the molar volume $\tilde{v}_m$ of the gas mixture, the real volumetric flow could be calculated,

[2] Assuming a homogeneous temperature distribution outside the pipes, and considering the high temperature of 1600 K, the outer heat transfer resistance was regarded as negligible.

with which the real flow velocity of the gas in the pipes could be obtained. The results are shown in Table 4.

$$\dot{V}_{real} = F_{tot} \sum_i \tilde{v}_m \cdot x_i \qquad \rightarrow \qquad u_{real} = \frac{\dot{V}_{real}}{A_{cross}} \tag{47}$$

Table 1: Critical temperature T_{cr} and pressure p_{cr} for each component found in the literature [8].

Component	T_{cr} **[°C]**	p_{cr} **[bar]**
CH_4	-82.59	45.99
H_2	-240.01	12.96
N_2	-146.96	33.96
HCN	183.5	5390.49
NH_3	132.25	113.3

2.1.3 Energy and Cost calculations

With the above optimized feed rate, all the feed streams can be calculated as well as the energy needed to preheat the feed from room temperature to 700 K and the reaction mixture from 700 K to 1600 K. From thermodynamics it is known that at constant pressure the produced heat is equal to the enthalpy.

$$dQ = dH \tag{48}$$

For an ideal gas, dH is given by:

$$dH^* = C_p dT \tag{49}$$

The feed stream has to be preheated from room temperature (298 K) to 700 K, which leads to the expression:

$$\Delta H_{heat} = \sum F_i^{out} C_p^{out} T^{out} - F_i^{in} C_p^{in} T^{in} \tag{50}$$

To heat up the reactor, the reaction enthalpies play a role in this calculation which gives equation (51).

$$\Delta H = \sum F_i^{out} C_p^{out} T^{out} - F_i^{in} C_p^{in} T^{in} + \sum r_i \Delta_r H \tag{51}$$

The outlet stream of the reactor (1600 K) was used to preheat the before mentioned feed stream. This energy can be transferred by using a heat exchanger. The temperature of the reaction gas after the heat exchanger was calculated to be $T_{rxn\ gas} = 1'316\ K$ by solving the following energy balance for T:

$$\Delta H_{heat}^{feed} - \Delta H_{cool}^{rxn\ gas}(T) = 0 \tag{52}$$

The reaction gas had to be cooled to the inlet temperature of the ammonia absorber (363 K). An initial temperature of the cooling water of 288 K was assumed. The temperature of the water should only be increased by 5 K, so that it would be possible to recycle the cooling water directly to the source (f. ex. river). By using these assumptions, the cooling water flow rate was calculated by using the following expression:

$$\Delta \dot{m}_w = \frac{H_{cool}^{rxn\ gas}}{C_{P,H_2O}(T_{out} - T_{in})} \tag{53}$$

Since the temperature range is only 5 K, the temperature dependence of the heat capacity of the cooling water can be neglected.
The optimization of the yield led to a minimum amount of reactors that had to be used to produce 10'000 kt/year. Considering the fact that every year, 30% of the reactor tubes have to be replaced, the capital costs for the reactors were calculated. Additionally, the expenses of the heat exchangers (preheater and cooler) were determined by the overall area needed to transfer the energy. This value was calculated using the general heat transfer equation:

$$A = \frac{P}{\alpha \Delta T} \tag{54}$$

with P being the power of the cooling/heating process, α the heat tranfer coefficient of the heat exchanger and ΔT the temperature difference that has to be reached with the corresponding power.

2.2 Calculations NH_3 Absorber

The gaseous product stream leaves the reactor and is fed into the absorption column, where the ammonia concentration is reduced to below 100 ppm. This specification arises from the danger of a highly exothermic polymerization reaction of prussic acid to occur, where ammonia acts as a homogeneous catalyst. The NH_3 absorption is achieved in a packed column using a liquid stream of 10 mol-% sulfuric acid in water (gas-liquid absorption). The ammonia reacts with sulfuric acid to form ammonium sulfate (cf. equation (55)), which can

be easily removed in solid phase. At sulfuric acid concentrations lower than 10 mol-% the absorption would be less efficient, whereas higher concentrations are unnecessary and would lead to corrosion, increasing the material costs.

$$2NH_3 + H_2SO_4 \longrightarrow (NH_4)_2SO_4 \tag{55}$$

2.2.1 Thermodynamic Analysis

In order to calculate the absorber temperature, an energy balance was set up:

$$\frac{dE}{dt} = F_{in} \cdot h_{in} - F_{out} \cdot h_{out} = \sum (F_{i,in} \cdot h_{i,in}) - \sum (F_{i,out} \cdot h_{i,out}) \tag{56}$$

The outlet flow rate can be described as a function of the inlet flow rate and the reaction term, so that the energy balance can be described by the inlet flow only:

$$\frac{dN_i}{dt} = F_{i,in} - F_{i,out} + r_i V = 0 \tag{57}$$

$$\frac{dE}{dt} = \sum F_{i,in} \cdot (h_{i,in} - h_{i,out}) - r \cdot V \cdot \Delta H_r \tag{58}$$

By assuming steady state, the energy balance can be set to 0 and the corresponding temperature can thus be calculated. The inlet flow consists of the components hydrogen cyanide, methane, ammonia, nitrogen and hydrogen in the gaseous phase and an aqueous solution of sulfuric acid in the liquid phase. In the outlet stream, there is an additional component, ammonium sulfate, in the liquid phase due to the reaction of ammonia with sulfuric acid. Equation (58) can then be rewritten as:

$$\begin{aligned}
\frac{dE}{dt} &= G_{in} \cdot (x^0_{NH_3} * \Delta h_{NH_3} + x^0_{CH_4} * \Delta h_{CH_4} + x^0_{HCN} * \Delta h_{HCN} + x^0_{H_2} * \Delta h_{H_2} + x^0_{N_2} * \Delta h_{N_2}) \\
&\quad + L_{in} \cdot (y^0_{H_2SO_4} * \Delta h_{H_2SO_4} + y^0_{H_2O} * \Delta h_{H_2O}) \\
&\quad - G_{out} \cdot ((x_{NH_3} * \Delta h_{NH_3} + x_{CH_4} * \Delta h_{CH_4} + x_{HCN} * \Delta h_{HCN} + x_{H_2} * \Delta h_{H_2} + x_{N_2} * \Delta h_{N_2}) \\
&\quad - L_{out} \cdot (y_{H_2SO_4} * \Delta h_{H_2SO_4} + y_{H_2O} * \Delta h_{H_2O} + y_{(NH_4)_2SO_4} * \Delta h_{(NH_4)_2SO_4}) \\
&\quad + \Delta H_r \cdot G_{out} \cdot (x^0_{NH_3} - x_{NH_3})/2 \\
&\quad = 0
\end{aligned} \tag{59}$$

Here, Δh_i represents the difference between the standard formation enthalpy (at T=298 K and 1 atm) and the formation enthalpy at a certain temperature T for component i. It can be used instead of the enthalpy itself, because the change of inlet and outlet flow rates as well as the molar fractions is very small, so that the standard reference formation enthalpies cancel out. Since the heat capacity is temperature-dependent in the operating temperature interval, it cannot be assumed to be constant. Therefore, the enthalpy difference must be approximated using the Shomate equation (61), where the parameters $A - D$, which are substance properties, are tabulated in the appendix and t is equal to $\frac{T[K]}{1000}$. x_i and y_i represent the molar fractions of each component in the gas and liquid phase, respectively. The molar fractions of the outlet gas and liquid stream can be obtained from the specification that no more than 100 ppm NH_3 is allowed in the gas stream after the absorption. By integration of equation (61) and using equation (60), an expression for the enthalpy difference was found (equation (62)).

$$h_f = h^0_f + \int_{T_0}^{T} c_p(T)\,\mathrm{d}T \tag{60}$$

$$c_p(T) = A + B \cdot t + C \cdot t^2 + D \cdot t^3 \tag{61}$$

$$h_f - h_f^0 = A \cdot t + \frac{B \cdot t^2}{2} + \frac{C \cdot t^3}{3} + \frac{D \cdot t^4}{4} \tag{62}$$

To compare the temperature difference of the non-ideal and ideal case, the calculation was also done assuming that all heat capacities are independent of temperature. This leaves only the parameter A in equation (61), such that the expression describing the enthalpy difference simplifies to:

$$h_f - h_f^0 = A \cdot t \tag{63}$$

2.2.2 Absorber Dimensions

The height of one transfer unit HTU can be determined using the following formula:

$$HTU = HG + \frac{m}{L/G} \cdot HL = HG \tag{64}$$

Since acid-base reactions are very fast, the reaction of ammonia with sulfuric acid can be considered instantaneous, so that the mass transfer in the liquid phase was neglected. Therefore, the height of one stage is controlled by the mass transfer in the gaseous phase only.

$$HG = \frac{G_{in}}{A_{cross} \cdot K_G \cdot a} \tag{65}$$

A_{cross} is the cross section of the absorption column, P the pressure inside the column (1.013 bar), K_G the overall mass transfer coefficient and a the total surface (190 m^2/m^3). K_G was obtained from the mass transfer correlation equation (66) [9], where G^{mass} is the mass flow rate, μ the dynamic viscosity, ν the kinematic viscosity, D the diffusion coefficient of ammonia in hydrogen and a_p the given packing factor (492 m^{-1}). To obtain the overall mass transfer coefficient of the gas phase K_G, the local mass transfer coefficient k_G had to be converted with equation (67) [9]. Since the gas flow mainly consists of hydrogen, all other substances were neglected for simplification. The diffusion coefficient could be approximated using formula (68), where the molar volumes were obtained from empirical correlations [9]. For calculating the dynamic viscosity, experimental data was fitted with a polynomial function of second order [4]. The temperature-dependent density could be calculated from equation (69), so that the kinematic viscosity is then given by equation (70).

$$k_G = 5.23 \cdot \frac{A_c \cdot a_p \cdot D}{R \cdot T} \cdot \left(\frac{G^{mass}}{a_p \cdot \mu}\right)^{0.7} \cdot \left(\frac{\nu}{D}\right)^{1/3} \cdot (a \cdot d)^{-2} \cdot P \tag{66}$$

$$K_G = \frac{k_G}{RT} \tag{67}$$

$$D = 10^{-3} \cdot \frac{T^{1.75} \cdot \left(\frac{1}{M_{H_2}} + \frac{1}{M_{NH_3}}\right)^{1/2}}{\left(V_{mH_2}^{1/3} + V_{mNH_3}^{1/3}\right)^2} \tag{68}$$

$$\rho = \frac{P \cdot M_{H_2}}{R \cdot T} \tag{69}$$

$$\nu = \rho \cdot \mu \tag{70}$$

The number of transfer units NTU is given by:

$$NTU = ln\left(\frac{x^0_{NH_3}}{x_{NH_3}}\right) \tag{71}$$

Therefore, the total height of the absorber is equal to the product of NTU and HTU. An additional height of $0.5\ m$ has to be taken into account for ground clearance.

$$h = NTU \cdot HTU + 0.5m \tag{72}$$

2.2.3 Cost Calculation

From an economic perspective, several points have to be included into the cost calculation. First of all, one has to include the cost of the absorption column itself whose price depends largely on its size (equation (73)). As the height of the column depends on the temperature, one has to optimize the liquid flow rate accordingly. Additionally, one should keep the latter as low as possible, as the costs for water, sulfuric acid and waste water increase with increasing flow rates.

$$\text{Cost of packed column} = 70000 \cdot (H \cdot D)^{1/2}\ [USD] \tag{73}$$

2.3 Calculations HCN Absorber

The gas stream leaving the NH_3 absorber is fed into a second absorber, which serves to purify the product HCN. Liquid water ($20°C$ and 1 atm) is used to absorb the prussic acid in a packed column, with a specification of less than 100 ppm product in the outlet gas flow. The approach to this problem was similar to that of the first absorber.
Similarly to the NH_3 absorber, the temperature of the second absorber was calculated from the energy balance. The only difference is that in this case, the reaction term does not exist. The prussic acid of the inlet gas stream is absorbed by a liquid stream of pure water at $20°C$. The resulting energy balance is therefore:

$$\begin{aligned}\frac{dE}{dt} &= G_{in} \cdot (x^0_{NH_3} * \Delta h_{NH_3} + x^0_{CH_4} * \Delta h_{CH_4} + x^0_{HCN} * \Delta h_{HCN} + x^0_{H_2} * \Delta h_{H_2} + x^0_{N_2} * \Delta h_{N_2}) \\ &\quad + L_{in} \cdot \Delta h_{H_2O} \\ &\quad - G_{out} \cdot ((x_{NH_3} * \Delta h_{NH_3} + x_{CH_4} * \Delta h_{CH_4} + x_{HCN} * \Delta h_{HCN} + x_{H_2} * \Delta h_{H_2} + x_{N_2} * \Delta h_{N_2}) \\ &\quad - L_{out} \cdot (y_{HCN} * \Delta h_{HCN} + y_{H_2O} * \Delta h_{H_2O}) \\ &\quad = 0\end{aligned} \tag{74}$$

The molar fractions of the outlet flows can be determined from the specification that the outlet gas stream should not contain more than 100 ppm HCN. Again, the calculation was done for the ideal case, where all heat capacity coefficients were assumed temperature-independent (cf. equation (63)), and the non-ideal case, for which the Shomate equation was made use of (cf. equation (62)). One main difference compared to absorber 1 is that since there is no instantaneous reaction in the liquid phase, the prussic acid concentration cannot be neglected. Thus, the complete form of equation 64 has to be used to obtain HTU. Moreover, for the calculation of HL the mass transfer coefficient in the liquid phase K_L was approximated using the correlation (76) and (77) [9]. ρ is the density, μ is the dynamic viscosity, ν the kinematic viscosity and M the molar mass of water. g stands for the gravitational acceleration. The diffusion coefficient of prussic acid in water was estimated using the Wilke-Chang correlation (78).

$$HL = \frac{L_{in}}{A_{cross} \cdot K_L \cdot a} \tag{75}$$

$$k_L = 0.0051 \cdot \left(\frac{\rho_{H_2O}}{\mu_{H_2O} \cdot g}\right)^{-1/3} \cdot \left(\frac{L_m}{a_p \cdot \mu_{H_2O}}\right)^{2/3} \cdot \left(\frac{\nu_{H_2O}}{D}\right)^{-1/2} \cdot \left(a_p \cdot d\right)^{0.4} \cdot \frac{M_{H_2O}}{\rho_{H_2O}} \tag{76}$$

$$K_L = \frac{k_L \cdot \rho_{fluid}}{M_{fluid}} \tag{77}$$

$$D = 4.4 \cdot 10^{-8} \cdot \frac{T}{\mu_{H_2O}} \tag{78}$$

Lastly, the other main difference to the first absorber is that the number of stages has to be calculated as follows:

$$NTU = \frac{(y_{i,in} - y_{i,out})}{\frac{(y_{i,in} - y^G_{i,in}) - (y_{i,out} - y^G_{i,out})}{ln\left(\frac{y_{i,in} - y^G_{i,in}}{y_{i,out} - y^G_{i,out}}\right)}} \tag{79}$$

This is due to the fact that compared to absorber 1, there is an equilibrium between the vapour and liquid phase. The equilibrium concentrations y^G_{in} and y^G_{out} of HCN are given by:

$$y^G_i = \frac{H}{P} \cdot x_i \tag{80}$$

2.3.1 Cost calculation

The costs for the HCN absorber consist of the expenses for the reactor and the needed process water as well as the revenues from the sold H_2. The costs for the absorber are calculated using the equation (73). The sales price for hydrogen was set to 1'000 USD per ton [10]. With an average production of 3.128 tons per year, a revenue of 3.1 Mio. USD per year is gained.

2.4 Calculations Distillation Column

After being absorbed by water, the hydrogen cyanide (HCN) must then be distilled and collected at a satisfactory purity for sale. This is done using a distillation column. In a distillation column a more volatile fluid is separated from a less volatile one. In this scenario, the more volatile compound is HCN, while the less volatile one is water. For the purpose of the simulation, it was assumed that the feed contains only these two compounds. The specifications for the distillation were given as follows:

Table 2: Given specifications for the distillation of hydrogen cyanide from water.

Purity of HCN in distillate (x_D)	0.995
Mole fraction HCN in bottoms (x_B)	1×10^{-5}

The mass balances over the distillation column are expressed as follows[11]:

$$F = D + B \tag{81}$$

$$Fz_F = Dx_D + Bx_B \tag{82}$$

z_F is the mole fraction of HCN in the feed, F is the molar flow rate of the feed, and D and B those of the distillate and bottoms respectively. Using these relations is it possible to calculate the flow rates of the distillate and bottoms if the feed is known.

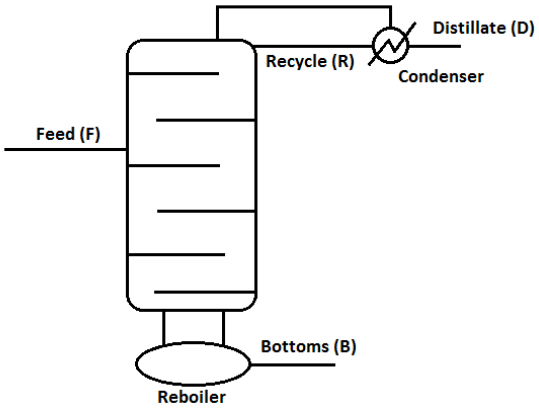

Figure 2: Schematic diagram of a distillation column, showing stages within the column, the feed, bottoms, distillate and recycle as well as the reboiler and condenser.

The Mccabe Thiele graphical method was chosen to calculate the column dimensions. This method was chosen due to the fact that it had already been taught in the lecture on separation processes. The first step in simulating the distillation column using the Mccabe Thiele graphical method involved choosing a thermodynamic model with which to calculate the equilibrium conditions. In order to make comparisons, two models were chosen. Naturally, the ideal mixture model was used, based on Raoult's law[11]:

$$y_i = \frac{P_i^s(T)}{P} x_i \gamma_i \tag{83}$$

where y_i and x_i represent the mole fractions of substance i in the vapour phase and liquid phases respectively, $P_i^s(T)$ is the saturation pressure of i, and P is the pressure in the system. In the ideal mixing case, the activity coefficients γ_i are equal to 1. In the non-ideal case, a model must be used. The model chosen is the van Laar model, and the activity coefficients are described as such [12]:

$$ln(\gamma_i) = A_{ij}\left(\frac{A_{ji}x_j}{A_{ij}x_i + A_{ji}x_j}\right)^2 \tag{84}$$

The van Laar coefficients A_{ij} and A_{ji} are obtained experimentally. Once the model had been applied, an equilibrium curve could be calculated for the compositions of the liquid and vapour phases.
The second step was to calculate the feed line. This line represents the feed, and has the following equation[11]:

$$y = \frac{q}{q-1}x - \frac{z_F}{q-1} \tag{85}$$

where q is the feed quality. This parameter is equal to the fraction of the feed which is in the liquid state. Since the feed enters the column at it's boiling temperature, the feed quality can be adjusted by adding or removing heat energy from the system, essentially condensing of vapourising it as required. The feed line starts on the diagonal at the composition of the feed, and intersects the equilibrium line at a certain point.
Once the feed line has been calculated, the upper operating line can be calculated graphically. This is done by drawing a line from the desired concentration in the distillate on the diagonal to the intercept of the feed and equilibrium lines. From the y-intercept of this line, the minimum reflux ratio can be calculated. The reflux ratio is the ratio of the recycle stream to the distillate stream. Once the minimum has been obtained, the ratio can be optimised. The equation for the upper operating line is as follows[11]:

$$y = \frac{RR}{RR+1}x + \frac{x_D}{RR+1} \tag{86}$$

where RR is the reflux ratio. Once the reflux ratio has been optimised, the operating line no longer intercepts the equilibrium line. In the last step, the lower operating line is calculated. This line begins at the point along the diagonal which denotes the concentration in the bottoms (x_D), until the intercept of the feed and upper operating lines. The equation is as follows[11]:

$$y = \frac{V_b + 1}{V_b}x + \frac{x_B}{V_b} \tag{87}$$

where V_b is the boil-up ratio. This is the ratio of liquid which is boiled in the reboiler to the liquid which is extracted as bottoms.

Once the equilibrium line and the two operating lines have been plotted, calculating the theoretical number of trays is done graphically in the x-y plane. The desired concentration in the bottoms (x_B) is chosen as a starting point on the diagonal. A vertical line is drawn to the equilibrium line, and then a horizontal line to the lower operating line. This is repeated until the point x_D on the diagonal. At the point where the lower operating line ends, the pattern is continued on the upper operating line. The number of steps is then the theoretical number of stages. Since equilibrium is not necessarily reached at each tray, the efficiency for the trays was assumed to be 0.7 [13]. The actual tray number is then [13]:

$$N_{actual} = \frac{N_{theoretical}}{\text{Efficiency}} \tag{88}$$

Once the actual tray number is known, the height of each tray is estimated as 0.732 metres [13] and the height was calculated. In order to calculate the diameter of the column, the maximum volumetric flow and the velocity of the vapour in the column is required. The maximum volumetric flow is taken for the tray directly above the feed. The molar vapour flow above the feed is calculated as follows:

$$V_{feed} = BV_b + F(1 - q) \tag{89}$$

which is the vapour flow from the reboiler plus the vapour fraction of the feed. This position was chosen as it is in the rectifying section, where the vapour load is highest. Next, the molar volume of a gas at 62.5^{o}C was calculated using the idea gas law [14]:

$$V_m = \frac{RT}{P} \tag{90}$$

An ideal gas was assumed due to the temperature being slightly elevated and the pressure atmospheric. The temperature of 62.5^{o}C was assumed as being the temperature near the feed stage, since it is the average of the boiling points of water and HCN. The maximum volumetric flow is then given as:

$$Q_{max} = V_m V_{feed} \tag{91}$$

The maximum vapour velocity can be calculated using the empirical relationship[13]:

$$v_{max} = \frac{1}{\sqrt{\rho_v}} \tag{92}$$

where ρ_v is the vapour density. For this calculation it was assumed that the vapour is made entirely of water, and the density of steam at 1 bar was used. Lastly, the cross-sectional area of the column was calculated:

$$A = \frac{Q_{max}}{v_{max}} \tag{93}$$

and using the equation for the area of a circle $A = \pi r^2$, the diameter was calculated.

Once the column dimensions have been calculated, it is necessary to calculate the power required to cool in the condenser and heat in the reboiler. For these calculations, it was assumed that the reboiler only needs to

deliver enough heat energy to vapourise the fraction of water that travels up the column. For the condenser, it was assumed that all the HCN needs to be liquefied. The powers were calculated as follows:

$$P_{condenser} = D\Delta H_{vap}(HCN) \tag{94}$$

$$P_{reboiler} = V_{up}\Delta H_{vap}(H_2O) \tag{95}$$

V_up is the boil-up ratio (V_b) multiplied by the bottoms (B), giving the flow of water vapour upwards from the reboiler. The heat transfer coefficients were used to calculate the required surface areas of the heat exchangers[15]:

$$A = \frac{P}{h\Delta T} \tag{96}$$

where P is the required power, h is the heat transfer coefficient, and ΔT is the average temperature difference between the cooling/heating medium and the substance to be cooled/heated. For the reboiler it was assumed that steam at 5 bar would be used for heating. The mass flow of steam to the reboiler was calculated as such:

$$\dot{m}_{steam} = \frac{P}{\Delta H_{vap}^{5bar}(steam)} \tag{97}$$

15% concentrated brine was assumed as a cooling medium in the condenser. It was assumed that the brine is fed at -10^{o}C, and that it does not warm up above 6 ^{o}C since the HCN condenses at 26 ^{o}C, and the heat exchanger has a differential temperature of 20 K.

$$\dot{m}_{brine} = \frac{P}{C_{p,brine}(T_{brine,out} - T_{brine,in})} \tag{98}$$

At this point the entire column has been calculated. Finally, the costs for the column must be calculated. The following relation was used to calculate the cost of the column[16]:

$$Cost_{Column} = 80320 \times Height^{0.76} Diameter^{1.21} \tag{99}$$

The cost of the heat exchangers was calculated using the following relation[16]:

$$Cost_{HeatExchangers} = 25000 \times HeatExchangeSurface^{0.65}; \tag{100}$$

The cost of steam was taken to be 25\$ per ton, and the cost of brine as 0.2\$ ton [17]. Since the distillation column and the HCN absorber need to be considered as a single unit, the first step in order to calculate their dimensions was to couple the two functions. This was done as described in the subsection on the HCN absorber. Since the bottoms may only contain 10 ppm HCN, it can be assumed pure. A least squares non-linear function optimised the water flow rate according to the total expenses over the 10 year depreciation period.

3 Results

3.1 Reactor

3.1.1 Optimization of the Yield

In the Degussa process hydrogen cyanide is produced from methane and a 5% excess of ammonia.

$$\mathrm{NH_3 + CH_4 \longrightarrow HCN + 3\,H_2} \qquad \Delta_r H = 251\ kJ/mol$$
$$\mathrm{2\,NH_3 \longrightarrow N_2 + 3\,H_2} \qquad \Delta_r H = 92.44\ kJ/mol$$

The reaction is highly endothermic with a reaction enthalpy $\Delta_r H$ of 251 kJ/mol [1]. The following experimental reaction rates for the production of HCN and the side reaction with the decomposition of ammonia to hydrogen and nitrogen were measured:

$$r_{HCN} = \frac{7.8 \cdot 10^{18}\ exp(-\frac{1950}{T}) p_{CH_4} p_{NH_3}^{0.5}}{(1 + 0.044\ exp(\frac{2390}{T})(p_{CH_4} p_{NH_3}^{-0.5}))^4} \tag{101}$$

$$r_{N_2} = \frac{4.9 \cdot 10^{18}\ exp(-\frac{2130}{T}) p_{NH_3}}{(1 + 0.044\ exp(\frac{2390}{T})(p_{CH_4} p_{NH_3}^{-0.5}))^3} \tag{102}$$

The reactor (length: 2 m, inner diameter: 15 mm and wall thickness: 2.5 mm), which is coated with sintered alumina is heated with natural gas. Platinum is used as a catalyst, which benefits the main reaction. The feed stream is preheated with a heat exchanger to 700 K and the outside temperature is not greater than 1600 K.
As a first step the feed rates of methane and ammonia were optimized with respect to the yield. Therefore, mass and energy balances were derived for a PFR (equation (103) and (104)).

$$\frac{dF}{dV} = r \cdot a \tag{103}$$

$$\frac{dT}{dV} = \frac{Ua(T_a - T) - \sum r_i \Delta_r H}{\sum F_i^{in} c_p} \tag{104}$$

U being the heat transfer coefficient of the pipe wall (later the overall heat transfer coefficient), T_a the heating temperature, r_i the reaction rate, $\Delta_r H$ the reaction enthalpy and c_p the heat capacity. Due to the fact that the temperature changes significantly, the heat capacity was adjusted by using the Shomate equation (14) [3]. By solving these ordinary differential equations a profile for the flows of each substance and a temperature profile as a function of the reactor length can be plotted. The optimal flow rate of the NH_3 inlet feed was calculated using the method of least squares in MATLAB to reach a maximum yield. The results are shown in Figure 3.

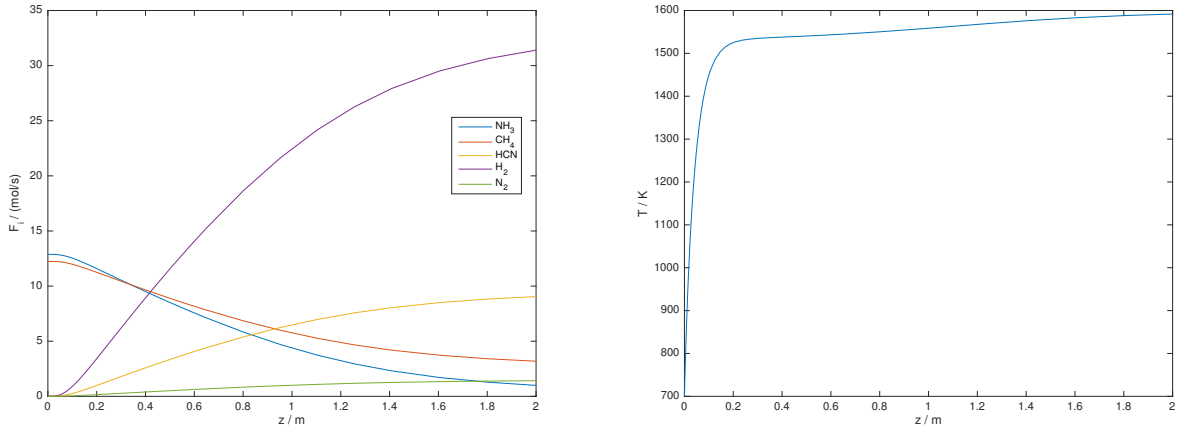

Figure 3: For the yield optimized flow rate profile over the reactor length (*left*) and the corresponding inner temperature profile (*right*) for an NH_3 inlet feed of 12.8 mol/s.

The optimised flow rate for NH_3 for one reactor was 12.8 mol/s where a yield of 74 % with respect to CH_4 was reached. It can be seen in Figure 13 in subsection 2.1 that the lower the flow rates, the higher the yields for HCN because more time is available for the reagents to react. However, almost all the reagents have already reacted at the beginning of the pipe (after 0.6 m) which makes the rest of the pipe underutilised. Therefore, a lower boundary was set in order to produce the desired 10'000 t/y for one reactor. This lower boundary leads to the required optimal reactant feed rate. In the temperature profile for a low NH_3 flow rate the desired temperature (1600 K) was achieved quickly. This is not the case for a high NH_3 flow rate which can be seen in Figure 12 in subsection 2.1. In this case 1600 K was not reached within the pipe. The production of HCN is very high in the latter case, however a lot of NH_3 and CH_4 have not reacted at the end of the pipe. This leads to a poor yield but can be made economically favourable through the use of absorbers after the reactor. The different outlet flow rates of HCN and the corresponding yields are combined in Figure 4.

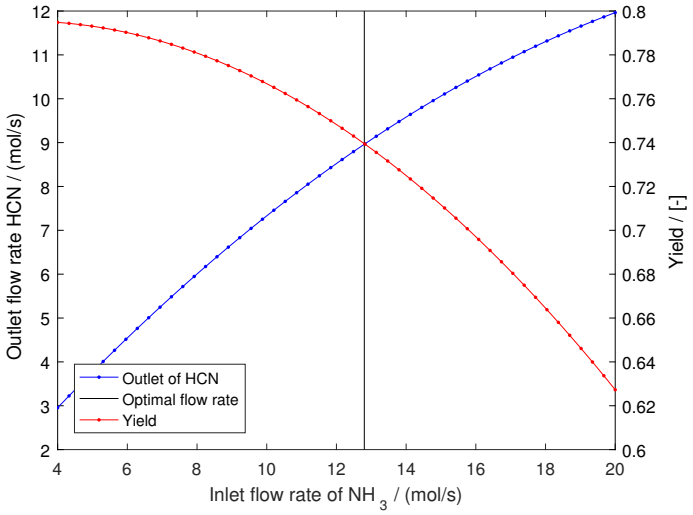

Figure 4: Shown is the relation between of the outlet flow rate of HCN and the corresponding yield with respect to methane. The optimal flow rate of 12.8 mol/s is indicated with a vertical black line. It is important to note that this analysis considers only the ideal behaviour of the gas mixture and the adaptation of the heat capacity to temperature changes.

3.1.2 Study of Parameter Dependence on Feed Rate

In the previous subsection some assumptions were made to simplify the modelling of the reactor and to get a first impression of its behaviour. Some of the parameters can be changed to make it more precise and their effects can be studied. One of them is the overall heat transfer coefficient U. It was assumed that only the resistance due to the wall plays a major role. However, the inner and outer gas flow both have a heat transfer coefficient, which need to be calculated in order to know whether or not they can be left out in further calculations (see equation (15))[3]. The heat transfer coefficient α_{in} was calculated using the Nusselt number which was obtained through a correlation with the Reynolds (equation (31)) and Prandtl number (equation (28)) [6]. This Nusselt number correlates to a tube with longitudinal flow and considers a change of the physical properties due to a wide temprature range:

$$Nu = 0.027 \cdot Re^{0.80} \cdot Pr^{1/3} \cdot (\mu/\mu_S)^{0.14} = \frac{\alpha \cdot d}{\lambda} \tag{105}$$

All three dimensionless numbers include parameters, such as the specific heat capacity C_p, the kinematic viscosity μ (μ_s is the viscosity at the highest temperature) and the thermal conductivity λ, which are specific for each temperature and composition. This means that they change over the reactor length. Therefore, they were first fitted for each species through a linear function in the case of the thermal conductivity and a quadratic

[3] Assuming a homogeneous temperature distribution outside the pipes, and considering the high temperature of 1600 K, the outer heat transfer resistance was regarded as negligible.

function in the case of the kinematic viscosity in order to get a temperature dependence [5]. For the specific heat capacity the temperature dependence was given through the Shomate equation (14) [3]. The heat capacity C_p, heat conductivity λ and dynamic viscosity μ for the reaction mixture were calculated under the assumption of an ideal mixture (equation (21) and (27)). They were then coupled with the ODEs and Nu and Re were calculated over the reactor tube (see Table 3 in subsection 2.1). The results (pressure and temperature profile of the optimized NH_3 inlet flow rate) are shown in Figure 5.

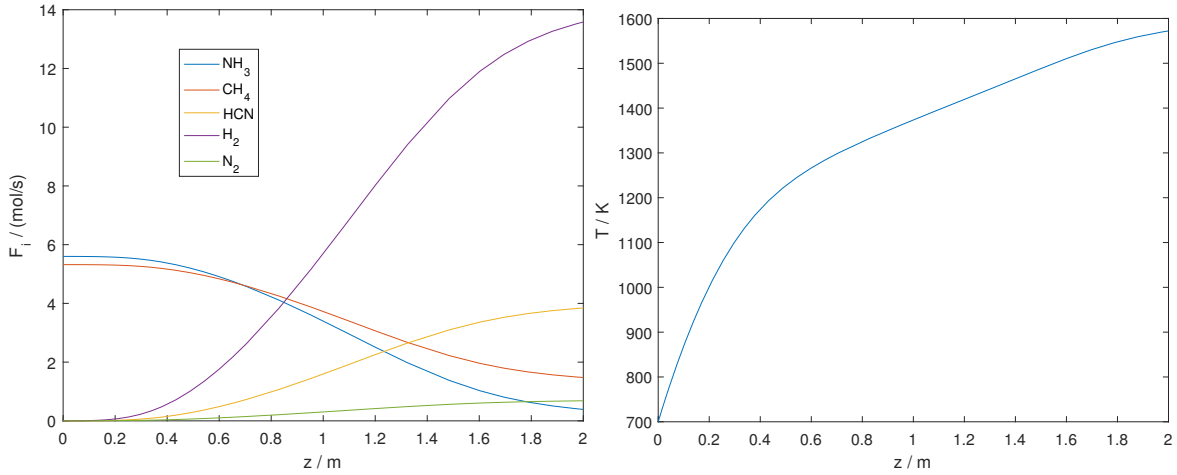

Figure 5: (*left*) The pressure profile of the species as a function of the reactor length is shown. The inlet flow rate of NH_3 was optimised (5.6 mol/s) to obtain a maximum yield of 69%. (*right*)

It can be seen that α_{in} indeed has a strong influence on the temperature profile as well as on the outlet streams. It takes longer to reach high temperatures. The reaction needs at least a temperature of 1'000 K to start, which means that some of the reactor length is only used to heat the reaction mixture and no reaction will occur in this subsection [1]. The HCN outlet stream is now much slower which means that less is produced at the end. In order to compensate the losses, the whole optimization has to be run again with the adjusted heat capacities, viscosities and the heat transfer coefficient. Since the yield decreases very much due to the adjusted temperature dependence of the reaction gas, one has to decrease the feed flow rate of the starting materials. In order to produce the desired 10'000 ton/year four reactors (instead of one), each with 400 tubes in it, were made available and the optimized inlet streams are changed to 5.6 mol/s NH_3 and 5.32 mol/s CH_4.

Table 3: Ranges of the Nusselt and Reynolds numbers as well as for the inner heat transfer coefficient α_{in} between the fluid and the pipe wall at the inlet (z = 0 m) and the outlet (z = 2 m) of the reactor.

Flowrate [mol/s]		**Reynolds [-]**	**Nusselt [-]**	α_{in} **[W/(m^{2}K)]**
0.0009	low flow	136 - 187	1.27 - 1.64	7.54 - 73.84
0.07	high flow	10'585 - 8177	41.40 - 33.68	245.50 - 832.12
0.014	optimized flow[a]	2117 - 2826	11.42 - 14.39	67.74 - 626.44

[a] Considering the adjusted temperature dependence of c_p, λ, μ and ν;

3.1.3 Sensitivity Analysis

With the newly calculated optimized flow rates a sensitivity analysis was conducted in order to get an even better yield based on the pressure in the pipe and the ratio of methane to ammonia.
The flow rates for HCN were plotted as a function of the reactor length at different pressures (from 1 atm up to 100 atm). The results are shown in Figure 6. It can be seen that the flow rate of HCN is maximised with

a pressure around 10 atm and decreases with higher ones rather quickly. This can be better seen on the right side of Figure 6 where the yield is plotted as a function of the pressure. The gain which comes with a higher pressure than the assumed 1 atm is very small compared to the enormous increase in energy consumption and costs. Therefore, the assumed 1 atm was near the maximum yield as visible in Figure 6. Due to this fact, $P = 1$ bar will be kept for further calculations.

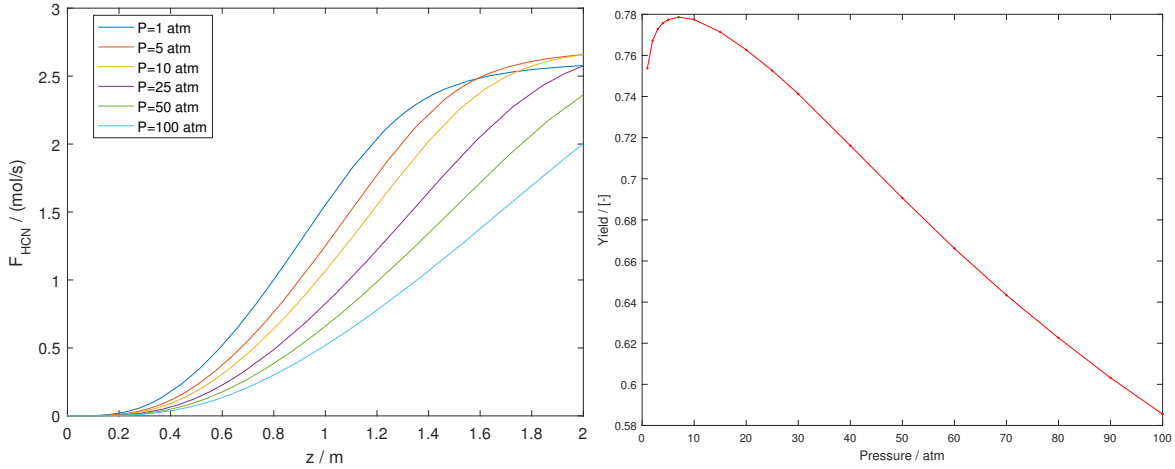

Figure 6: (*left*) The flow rates for HCN were plotted as a function of the reactor length at different pressures (from 1 atm up to 100 atm). The flow rate of HCN is maximised with a pressure around 8 atm and decreases with higher ones rather quickly. The gain which comes with a higher pressure than the assumed 1 atm is very small compared to the enormous increase in energy consumption and costs. (*right*) The same behaviour can be seen in the right plot where the yield is plotted as a function of the pressure.

As a second sensitivity analysis the ratio between methane and ammonia was changed and the results are shown in Figure 7.

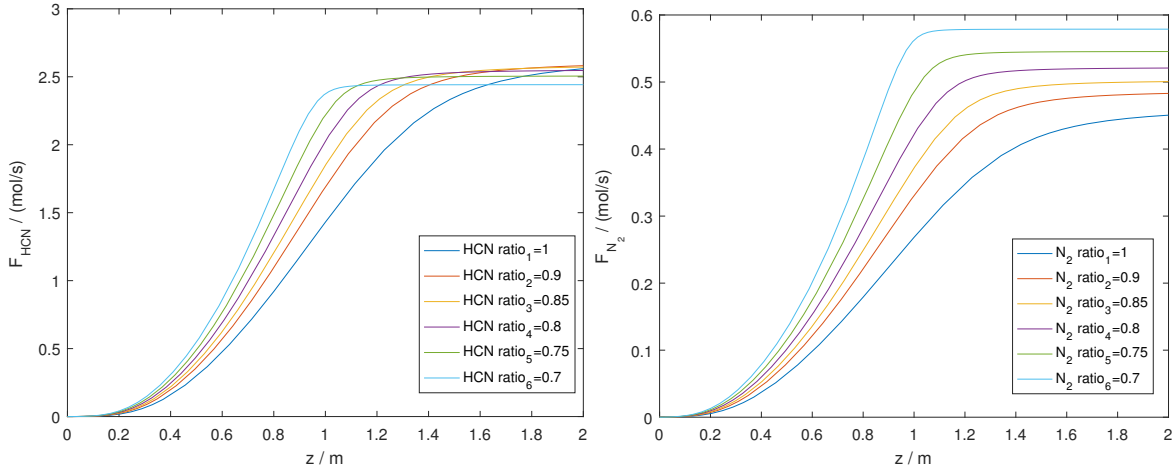

Figure 7: The HCN (*left*) and the N_2 flow rates (*right*) are shown as a function of the reactor length at different methane/ammonia ratios. There is again a trade-off between a higher HCN flow rate and a preferably small production of N_2.

At the beginning of the reactor it can be seen that the higher the excess of ammonia the faster the HCN flow rate increases. However, at the end of the reactor pipe an excess of ammonia of around 10% provides the highest HCN flow rate. Nonetheless, if the flow rate of N_2 which is formed during the side reaction is considered, one can see that an excess of ammonia automatically leads to a higher amount of this side product. Again,

it is a trade-off between a higher HCN flow rate and a preferably minimal production of N_2. Considering the higher production of N_2, which cannot be sold in the end, also in an economic point of view a higher methane/ammonia ratio would not be preferred.

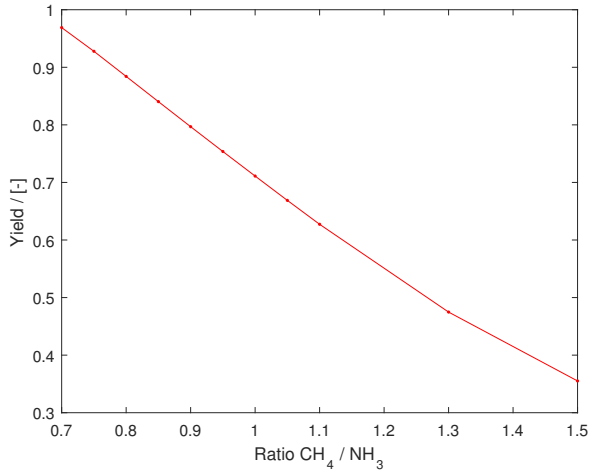

Figure 8: The yield is shown as a function of the methane/methane ratio. The yield increases almost linearly with a higher amount of ammonia.

In order to choose the optimal methane to ammonia ratio, the yield was plotted against the aforementioned ratio. Figure 8 shows that the yield increases linearly with a higher NH_3 feed rate. By considering all aspects, an excess between 5 to 10 % with respect to ammonia is preferred, which was already assumed for previous calculations. Therefore, the obtained temperature and composition profile which are shown in Figure 5 are considered valid.

3.1.4 Influence of the Thermodynamic Model of the Gas Phase

The mechanical pressure P was replaced by the fugacity f, in order to evaluate the impact of thermodynamic non-ideality on the operation of the reactor. The procedure of calculation is shown in detail in subsection 2.1.2. Since the compressibility factor of the gas mixture was determined to be $Z = 1$, the fugacity coefficient ϕ was calculated to be very close to unity ($\phi \approx 1$). The introduction of temperature dependence of the heat capacity, heat conductivity and kinematic and dynamic viscosities was therefore an important correction and made the consideration of fugacity redundant.
Lastly, the real flow velocity of the reaction gas was calculated. By rearranging the Peng-Robinson equation explicitly for the pressure (46)[7], the molar volume $\tilde{v}_m$ could be obtained and the real volumetric flow rate was calculated using equation (47). These calculations are shown in subsection 2.1.2. Similar to the ideal mechanical pressure, the ideal flow velocity in the pipe is not affected significantly.
There is a reasonable explanation for why the additional consideration of the fugacity and the real flow velocity do not make a significant difference for the calculations. The ideal gas law is most precise if a low pressure or a high temperature is considered. Since the reaction occurs at 1 atm and between 700 K and 1600 K, it can be considered as sufficient for the calculations. The adaptation of the heat capacity, heat conductivity and dynamic and kinematic viscosities to the temperature and the use of an inner heat transfer coefficient were therefore precise enough to model the reaction system.

3.1.5 Energy and Cost Calculation

Considering the cost for the natural gas to heat the reactor, the expenses for the starting materials and cooling water, the total expenses for the reactor system was determined. These calculations were based on the prices given in Table 5.

Table 4: Comparison of the real and ideal flow velocity at the inlet (z = 0m) and outlet (z = 2m) of the reactor pipes.

Velocity	Inlet [m/s]	Outlet [m/s]
u_{ideal}	8.8741	36.4502
u_{real}	8.8737	36.4558

Table 5: Prices for reactor and its maintenance, heat exchanger, cooling water for the reaction part and starting materials.

Process part	Unit	Price per unit	Total price for 10 years [USD]
Reactor[a]	4	$0.15 \cdot 10^6$ USD	$0.6 \cdot 10^6$
Maintenance[b]	30%	100 USD/pipe	$0.48 \cdot 10^6$
Heat exchanger[c](preheat)	1 with 3.38 m^2	55'200 USD	55'200
Heat exchanger[c](cooler)	1 with 4.08 m^2	62'400 USD	62'400
Natural gas[17]	0.0822 kg/s	0.25 USD/kg	$5.92 \cdot 10^6$
Cooling water	158 kg/s	0.1 USD/kg	$4.56 \cdot 10^6$
NH_3 feed[10]	0.38 kg/s	0.23 USD/kg	$25.2 \cdot 10^6$
CH_4 feed[10]	0.34 kg/s	0.25 USD/kg	$24.5 \cdot 10^6$
Total cost reaction			$61.5 \cdot 10^6$

[a] One reactor block contains 400 pipes; [b] Replacement of pipes per year; [c] Price depends on surface area;

3.2 NH_3 Absorber

The unreacted ammonia in the outlet reactor stream has to be removed to below 100 ppm. This is done by reaction of ammonia with sulfuric acid to ammonium sulfate, which can be easily removed. The liquid inlet stream is fed at a temperature of 20°C and a pressure of 1.013 bar, while the gas stream leaving the reactor enters the absorber at a temperature of 90°C. Since the total absorber temperature and the dimension of the absorption column depends on the feed streams as well as their molar composition, the latter can be varied to acquire an optimal temperature. The gaseous inlet stream is given by the outlet stream of the reactor and the aqueous sulfuric acid stream was calculated according to the number of moles of ammonia entering the column. The amount of sulfuric acid needed for the reaction was thereby determined and an equivalent of 1.4 with respect to ammonia was chosen. If the streams are assumed to be ideal, i.e. temperature independent, the heat capacities can be taken as constant for all species. In this case, a temperature of 57.45°C was obtained. When taking the thermodynamic non-ideality into account, one can use the Shomate equation to approximate the formation enthalpies at a certain temperature. With this method, the temperature was determined to be 32.34°C. One can see that for the assumption of ideal gases and liquids, the total absorber temperature is higher than if temperature dependence is factored into the energy balance. The optimal temperature for absorption is around 35°C. However, the reaction is favoured at high temperatures due to higher reaction constants (cf. Arrhenius equation (106)). A temperature above 100°C is undesirable as the feed medium would start boiling. Since this is not the case for neither the ideal nor the non-ideal matter, one does not have to enhance the liquid flow rate to lower the overall temperature.

$$k(T) = k \cdot \exp\left(-\frac{E_A}{RT}\right) \tag{106}$$

All in all, the height of one transfer unit was calculated to be 0.49 m and 1.74 m for the non-ideal and ideal case respectively. The number of transfer units is 6. Therefore a total column height of 3.46 m (non-ideal) and 10.94 m (ideal) was obtained, considering the ground clearance of 0.5 m.

3.2.1 Cost Analysis

The cost of the packed column depends on the column height and its diameter. It was observed that the column height increases with decreasing temperature due to the temperature dependence of the mass transfer coefficient. Furthermore, the temperature decreases with increasing liquid flow rate as the inlet temperature of the liquid is lower than that of the gas. As a consequence, the overall costs can be lowered by keeping the liquid flow as low as possible, while satisfying the upper limit of $100°C$ (boiling temperature of water at atmospheric pressure). The amount of water and sulfuric acid needed for the absorption is equal for both the ideal and non-ideal case. The only difference in costs lies in the column price. One can see that the temperature calculated in the ideal case is much higher than for non-ideality. Due to the increased height at higher temperature, the column cost is increased for the case of thermodynamic ideality as the column price is proportional to $H^{0.5}$ (cf. equation (73)). Ultimately, the total costs over 10 years for the non-ideal case amount to 2.65 mio. USD whereas they are 110'000 USD higher for the ideal case (2.76 mio. USD). The overall results are given in tables 6 and 7.

Table 6: Prices and total costs for NH_3 absorber with consideration of thermodynamic non-ideality

Process part	**Unit**	**Price per unit**	**Total price for 10 years [USD]**
HTU	0.49 m	–	–
NTU	6	–	–
Absorption column	1	148'000 USD	148'000
Sulfuric acid	0.39 ton/h (10 mol-%)	75 USD/ton	$2.31 \cdot 10^6$
Cooling water	0.64 ton/h	0.1 USD/ton	5'100
Waste water	1.12 ton/h	2 USD/ton	180'000
Total cost NH_3 absorption			$2.64 \cdot 10^6$
Temperature			$32.34°C$

Table 7: Prices and total costs for NH_3 absorber assuming thermodynamic ideality

Process part	**Unit**	**Price per unit**	**Total price for 10 years [USD]**
HTU	1.74 m	–	–
NTU	6	–	–
Absorption column	1	267'000 USD	267'000
Sulfuric acid	0.39 ton/h (10 mol-%)	75 USD/ton	$2.31 \cdot 10^6$
Cooling water	0.64 ton/h	0.1 USD/ton	5'100
Waste water	1.12 ton/h	2 USD /ton	180'000
Total cost NH_3 absorption			$2.76 \cdot 10^6$
Temperature			$57.45°C$

3.3 HCN Absorber

3.3.1 Results prior to the Coupling with the Distillation Column

Firstly, the temperature was determined using an energy balance. The temperature of the gaseous inlet stream was given as it is equal to the temperature obtained in absorber 1. The water gas stream was set to 1'500 kmol/h. Two different temperatures were obtained, an ideal and a non-ideal one, respectively. In the latter case the Shomate equation was used to calculate the heat capacities and the enthalpies. For the ideal case a temperature of 18.0°C was obtained whereas the non-ideal lay slightly higher at 21.7°C.
The height of the column was 2.90 m in the ideal model with 8 stages and 0.316 m between them. In the non-ideal case the unit is slightly taller, at 2.92 m. The gap between the two stages is therefore also larger, at 0.319 m, but the number of stages stays the same.
It can be seen that the difference between the ideal and non-ideal model is quite small. A gas mostly behaves ideally when the pressure is low or the temperature high[14]. Even though HCN is highly polar, one could deduce from this result that due to the low pressure it could behave ideally. Therefore, one can say that the ideal case is sufficient for modelling the HCN absorber. The outlet stream of HCN is 15.4 mol/s.

3.3.2 Results after the Coupling with the Distillation Column

Due to the fact that the water is almost pure after the distillation column (with an HCN concentration of less than 10 ppm), it can be used to absorb the prussic acid in the second absorber. Therefore, the distillation column is coupled with the HCN absorber through a recycling stream. The same equations as in the uncoupled model were used to calculate the absorber height and temperature. An algorithm was applied to minimize the costs by changing the inlet water flow. The temperature of the gaseous inlet stream is given again by the first absorber. The calculated temperatures for the ideal and non-ideal case are 9.8°C and 24.3°C, respectively.
The height of the column was determined to be 2.96 m, with an HTU value of 32.5 cm and a NTU of 8 for the ideal case. For the non-ideal case the height was a bit lower with 2.87 m and an HTU and NTU value of 31.3 cm and 8, respectively.
The difference in the dimensions of the column is therefore minimal in the ideal and non ideal cases, with only the temperature showing a significant difference between the two. This could be due to different inlet water streams coming from the distillation column. Therefore, differences between ideal and non-ideal are visible in the coupled system and not in the uncoupled, because in the latter case the liquid inlet streams are the same for ideal and non ideal.

3.3.3 Cost Analysis

The total expenses for the HCN absorber, which were calculated only after the coupling was completed, were 213'273 USD in the ideal case and 210'650 USD in the non-ideal case, respectively. Those costs depend strongly on the flow rate of water which is given by 298 kmol/h. It is important to note that the water flow rate is now not even a fifth from the one before the coupling. Therefore, one can conclude that the lower the flow rate the smaller the costs. Along with those expenses, a lot of revenues can be made by selling the produced H_2. The sales price for hydrogen was set to 1'000 USD per ton [10]. With an average production of 3'128 tons per year (average production rate of 54.3 mol/s), a revenue of 3.1 mio. USD per year is gained.

3.4 Distillation Column

3.4.1 Column Dimensions

The Mccabe Thiele method relies on a graphical analysis in order to deliver results. The following graph was obtained for the van Laar model.

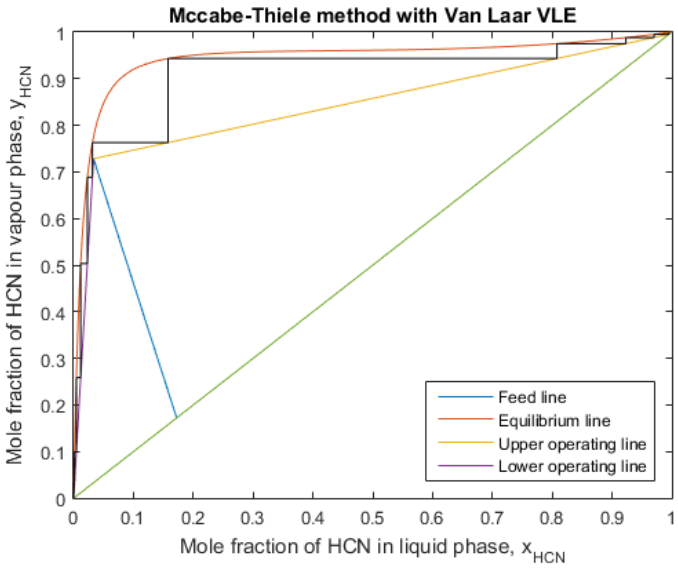

Figure 9: The Mccabe Thiele graph obtained for the van Laar model. Visible are the feed line, the upper operating line, the lower operating line, the diagonal, and the equilibrium line. The steps represent the theoretical stages in the column.

For comparison, the same graph corresponding to the ideal mixture model is shown below.

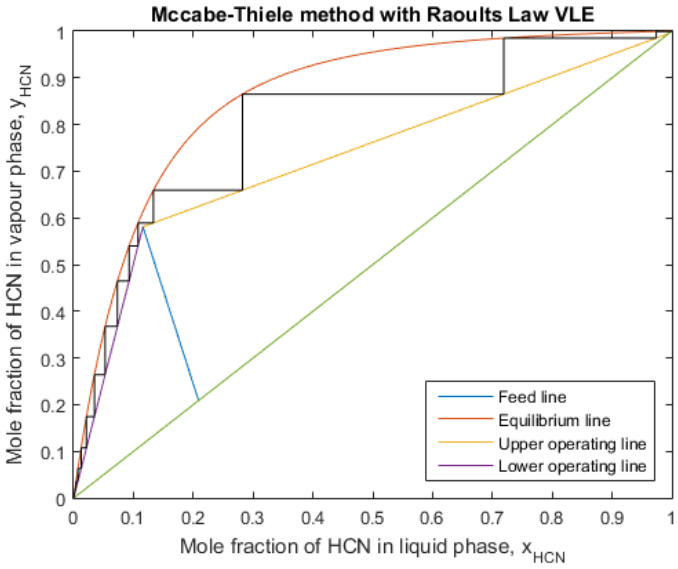

Figure 10: The Mccabe Thiele graph obtained for the ideal mixture model. The difference in the equilibrium line is clearly visible, which leads to highly differing results.

It appears that a mixture of HCN and water behaves in a highly non-ideal fashion. Purely comparing the equilibrium line of HCN in the gas and liquid phases (the red curves in figures 9 and 10), the difference is apparent. Additionally, the calculated boiling point of the feed mixture differs from 36.0°C in the non-ideal case, and 76°C

in the ideal one. Since the entire calculation of the column depends heavily on the thermodynamical model, the difference in the two models is magnified through the design process. It is therefore of utmost importance that an appropriate model be chosen when designing chemical processes.

	Ideal (Raoult)	Non-ideal (van Laar)
Height [m]	26.7	15.4
Diameter [m]	0.87	0.68
Stages	37	22
Feed stage	30	14
Reflux Ratio	1.22	0.384
Water recycle to HCN absorber [kmol h^{-1}]	298	298
Reboiler Area [m^2]	23.2	4.94
Condenser Area [m^2]	14.2	14.2
Operating costs [$] (10 years)	2'741'780	937'990
Capital costs [$]	1'152'440	615'987
Total costs [$]	3'894'210	1'553'980

Table 8: The dimensions of the distillation column as calculated using the Mccabe Thiele method. In one case, the ideal liquid vapour equilibrium model was used, in the other the van Laar model for the activity coefficients was applied. The differences are fairly large, due to the differences in the equilibrium models.

3.4.2 Reflux Ratio and Total Cost

Once the water flow rate in the two systems had been optimised, the reflux ratio could be optimised. This was done by plotting the total expenses for the distillation unit as a function of the reflux parameter. The reflux parameter is the multiplication factor of the minimum reflux ratio. The optimisation was done for the van Laar vapour liquid equilibrium model: the same data was used to calculate the ideal case for comparison.

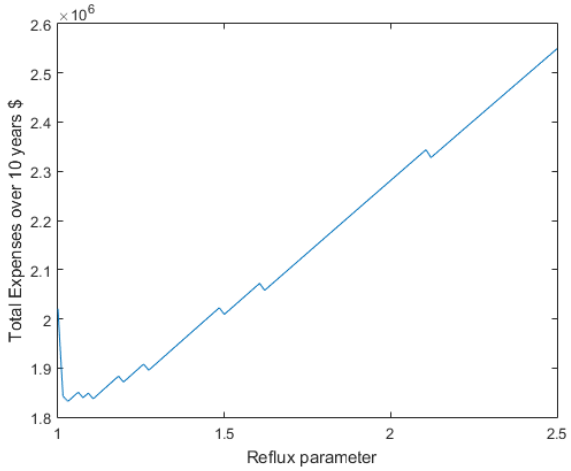

Figure 11: The total expenses for the distillation column expressed as a function of the reflux parameter. The data was calculated using the van Laar model, however a similar graph was obtained for the ideal model as well.

It was important to not only take the capital expenses into account, but the operating costs over the 10 year

period as well. As shown in the plot, the cost optimum reflux parameter occurs around 1 - 1.2. After this, the costs increase mostly linearly. This comes from the fact that more HCN must be liquified in the condenser in order to keep the distillate rate constant. This not only incurs higher capital costs due to the larger condenser required, but also a larger amount of cooling medium.

It is important to note that the method used (Mccabe Thiele), is considered a shortcut method. This means that its use is primarily to obtain a rough idea of the size of the column required for a given separation. Additionally, due to the graphical nature of the method, it is difficult to obtain accurate results when the purities of the distillate and bottoms are required to be high. This is due to the possibility of pinch points, or areas where the equilibrium and operating lines are too close. This causes problems for the step algorithm used to calculate the number of stages. Additionally, a large number of assumptions were made during the design process, such as taking the average of the boiling points of the two substances as the temperature in the middle of the column. Naturally, these generate a certain error in the calculations. After using a shortcut method, more rigorous modelling needs to be done in order to further optimise operation parameters and reduce the capital and operating expenses.

4 Conclusion

The endothermic Degussa process (BMA) was used to produce hydrogen cyanide through methane and ammonia. A production plant with a production volume of 10'000 tons/year was simulated.
The purified and preheated reagents were sent to the four reactor units, each containing 400 tubes which are coated with platinum. The inlet flow rates were set to 5.32 mol/s for CH_4 and 5% in excess of ammonia was taken. With a conversion of 69% with respect to methane, a HCN outlet flow rate of 15.4 mol/s was received. Through a sensitivity analysis it was found that this flow rate can be maximised by setting the pressure to 10 atm in the reactor. However, the gain which comes with a higher pressure than the assumed 1 atm is very small compared to the enormous increase in energy consumption and costs. Therefore, the pressure was left at 1 atm. Similar results were received when changing the methane-ammonia ratio. The highest HCN flow rate was obtained with an ammonia excess of 10%. This in turn leads to a higher amount of produced N_2 through a side reaction. By considering all aspects of this trade-off, an ammonia excess of 5% was set. Additionally the heat transfer between reaction gas and the solid pipe was considered. Since there is a significant difference in heat transport compared to the ideal case, fluid-solid-heat transfer needs to be taken into account to model the reaction system in a more appropriate way. As a last step to consider non-ideality, the mechanical pressure was replaced by the fugacity using Peng-Robinson equation of state. However, the fugacity coefficient was very close to unity that no distinction to the mechanical pressure could be made. With the obtained results, the costs for the reactor were estimated. Considering expenses for raw materials, cooling and heating equipment inclusiding cooling water and natural gas and the reactor blocks, the overall costs for ten years are approximately 61.5 mio. USD.
After the reaction the gas stream is sent to an absorption column where the ammonia concentration is reduced to below 100 ppm. NH_3 reacts with sulfuric acid to form ammonium sulfate, which can be easily removed. The operating conditions and reactor dimensions were found through various calculations. The feed stream enters with a temperature of $20°C$ and leaves the column with $32.34°C$. The height of the absorber is 10.94 m with 6 transfer units and 1.74 m between them. Those results which where found by assuming non-ideality differ quite a bit from the ideal model. The economic aspect was also considered for this operational unit. Total expenses of 2.65 mio. USD were obtained over 10 years.
After the gas stream was free of NH_3, a second absorber was used to extract HCN with water. In order to optimize the costs for the distillation column, which is one of the major matters of expense, it was coupled with the HCN absorber through a water recycling system. After the optimisation was performed, the HCN absorber became 2.87 m high with 8 stages and 31.3 cm between them. The temperature in the absorber is $24.3°C$. During the optimisation process, the flow of water in the recycle was lowered to an optimum level. The expenses for the absorber are 210'650 USD. The revenues, however, were much higher, due to sales of hydrogen gas. With a sales price of 1'000 USD per ton and an average production of 3'128 tons per year, a profit of 3.1 mio. USD per year is obtained.
Once the HCN gas has been absorbed into water, it must be separated and purified. This is achieved by distillation. The results of the non-ideal van Laar model were considered to be most relevant. The distillation unit is 15.5 m tall, has a diameter of 0.68, 22 stages, and a reflux ratio of 0.384. The unit costs 615'987 USD to build and 937'990 USD to operate over 10 years. At the top, the distillate is 99.5% pure HCN, and the bottoms contains no more than 10 ppm of HCN. The bottoms is returned to the HCN absorber as wash water. The distillate flow rate is then 11.969 kt/a for HCN only. This is slightly above the given production target, however this limitation comes from the reactors. Each reactor provides a fixed amount of HCN, and therefore the target would have to be over or undershot.
As far as the applicability of the results obtained is concerned, there may very well be room for improvement. Many assumptions were made during the modelling process. However, most significantly, the methods used in some parts of the study were short cut methods, which themselves contain a number of assumptions and inaccuracies. The next step in such a process would be more rigorous modelling, during which optimisations and improvements could be made. With that being said, the results obtained can be considered within the

realm of the realistic, and could therefore serve as a good starting point for further modelling to take place.

5 Outlook

The outlook for the plant is extremely good in terms of profitability. The total cost for the plant is approximately 65.8 mio. USD, including operating costs over 10 years. The total revenue from the sale of HCN and H_2 gas is 178 mio., giving a profit of 112 mio. USD. This is a resoundingly positive result.
The environmental impact of the plant must also be taken into account. Nothing from the plant is being discharged into the environment. The bottoms water from the distillation is recycled and the ammonium sulfate is relatively safe, even if small amounts remain dissolved in the water after filtration. There is always the chance of a larger disaster occurring, such as an explosion or a HCN leakage. However, chemical plants are generally built with a large number of safety precautions, reducing this risk greatly. Therefore, it is highly unlikely that there will be an environmental impact on the surrounding region.

References

[1] M. Bewersdorf, D. Wolf, J. Sauer, M. Köstner, M. Rinner in *Handbook of Heterogeneous Catalysis*, **2008**, pp. 2592–2609.

[2] M. Jakubith in, *Vol. 2*, Wiley-VCH Verlag GmbH & Co. KGaA, Weinheim, **1998**.

[3] National Institute of Standards and Technology, General References, Physicochemical Properties, **2016**, http://webbook.nist.gov/.

[4] Engineeringtoolbox, **2016**, http://www.engineeringtoolbox.com/.

[5] N. C. Blais, J. B. Mann, *The Journal of Chemical Physics* **1960**, *32*, 1459, http://scitation.aip.org/content/aip/journal/jcp/32/5/10.1063/1.1730942.

[6] R. K. Sinnott, J. Coulson, J. F. Richardson, *Chemical Engineering Design*, 3rd, **2005**.

[7] D. Jerinic, J. Schmidt, L. Friedel, *Chemie-Ingenieur-Technik* **2009**, *81*, 1397–1415.

[8] Physicochemical Properties of Gases, **2016**, http://encyclopedia.airliquide.com/Encyclopedia.asp?GasID=2%7B%5C#%7DGeneralData.

[9] E. L. Cussler, *Diffusion - Mass Transfer in Fluid Systems*, 2nd, **2009**, pp. 117–160.

[10] ICIS, Raw Material Prices, **2016**, http://www.icis.com/chemicals/channel-info-chemicals-a-z/.

[11] M. Mazzotti, Hyper-Thermische-Verfahrenstechnik: Interaktive Lerneinheit, **2016**, http://www.hyper-tvt.ethz.ch/.

[12] D.-Y. Peng, *The Open Thermodynamics Journal* **2010**, *4*, 129–140.

[13] A. Linninger, Distillation, **2003**.

[14] A. Krönig, *Annalen der Physik* **1856**, *175*, 315–322.

[15] MIT Web Lessons, **2016**, http://web.mit.edu/16.unified/www/FALL/thermodynamics/notes/node131.html.

[16] Matches, Process Equipment, **2014**, http://www.matche.com/equipcost/EquipmentIndex.html.

[17] Index Mundi, Energy Prices, **2016**, http://www.indexmundi.com/commodities/.

Appendix

A Additional Results Reactor Modelling

A.1 Optimization of the Yield

The molar flow rates for one pipe were plotted against the reactor length for different NH_3 feed rates. In Figure 13, a NH_3 feed rate of 4 mol/s was used, which is too small in order to produce the desired 10'000 t HCN/y with one reactor. In Figure 14, the yield was optimized using the method of least squares in MATLAB and a feed rate of 12.8 mol/s was found for NH_3. This result is due to a fixed lower boundary of the NH_3 feed rate in order to produce the right amount of HCN. In Figure 12 a too high NH_3 feed rate was chosen (20 mol/s). The amount of HCN produced is higher but the yield is smaller due to the lower conversion of CH_4. In the temperature profile in Figure 12 it is visible, that the desired 1600 K are never reached.

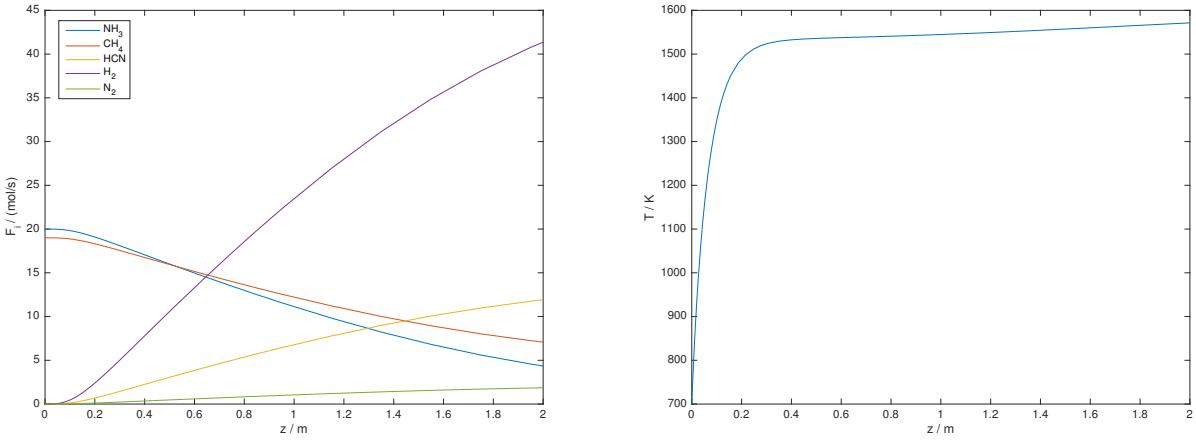

Figure 12: The flow rate was plotted against the reactor length for a very high NH_3 feed rate of 20 mol/s (*left*). The amount of HCN produced is higher but the yield is smaller due to the lower conversion of CH_4. In the temperature profile (*right*) it is visible, that the desired 1'600 K are never reached.

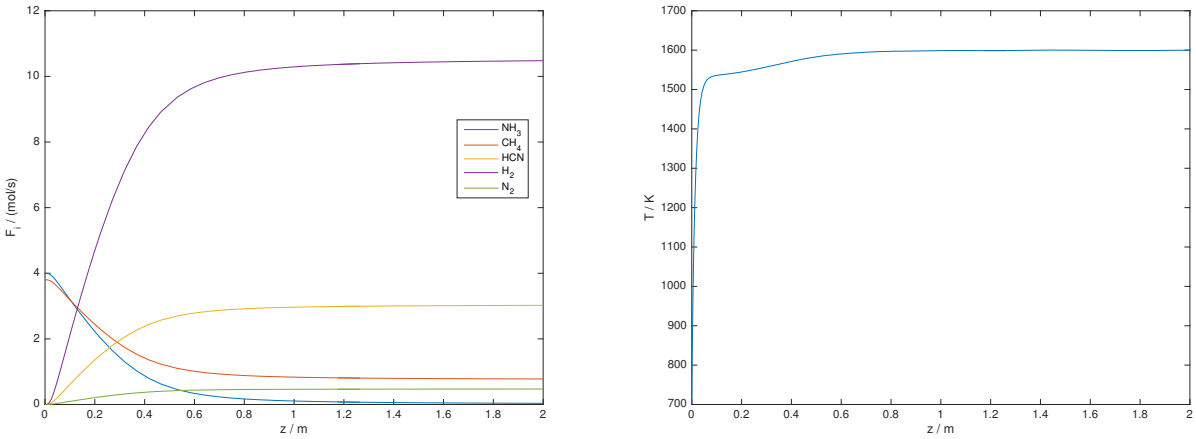

Figure 13: The flow rate was plotted against the reactor length for a very low NH_3 feed rate of 4 mol/s (*left*). It can be seen that the yield will be maximised in this way because almost all the CH_4 has reacted at the outlet of the reactor. However, this is already the case after a reactor length of 0.4 m which makes the rest of the reactor useless. Also the desired 10'000 t/y HCN will not be produced with only one reactor. On the right-hand side, the temperature profile for this NH_3 flow rate is shown.

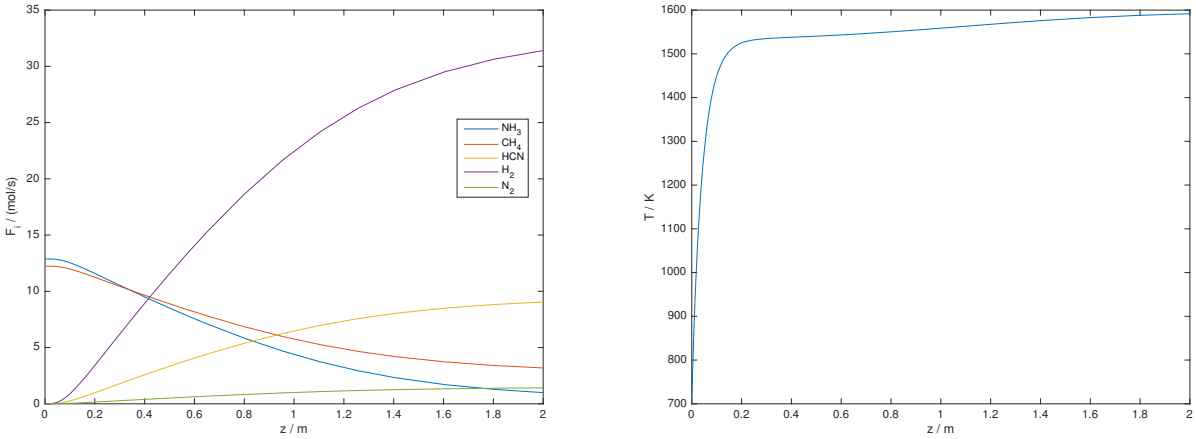

Figure 14: Shown are the flow rates- (*left*) and temperature (*right*) profile over the reactor length for an inlet feet of NH_3 that is optimized (12.8 mol/s) to reach a maximum yield.

A.2 Study of Parameter Dependence on Feed Rate and Influence of Other Thermodynamic Models of the Gas Phase

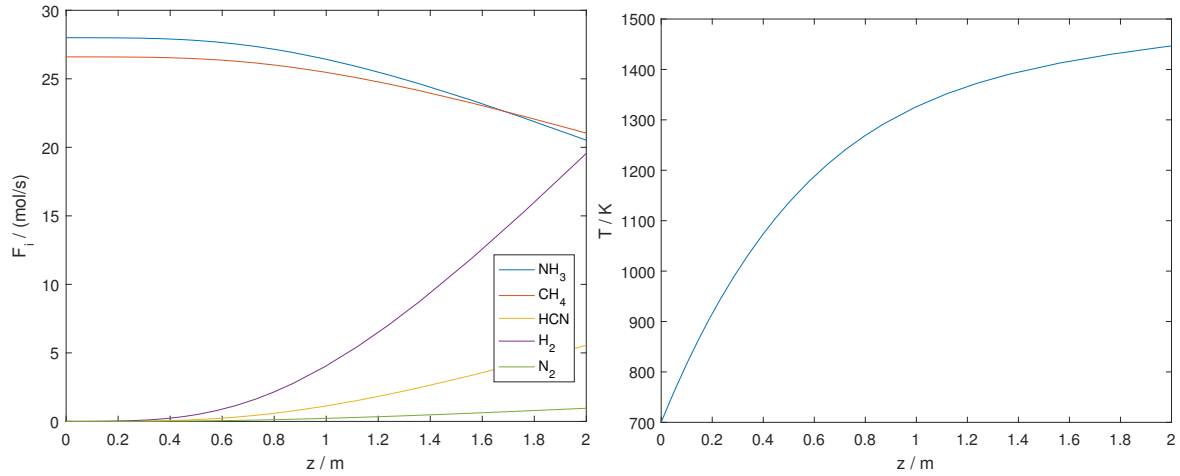

Figure 15: Flow rate- (*left*) and temperature (*right*) profile for the new obtained overall heat transfer coefficient and a very high NH_3 inlet flow rate of 28 mol/s are shown.

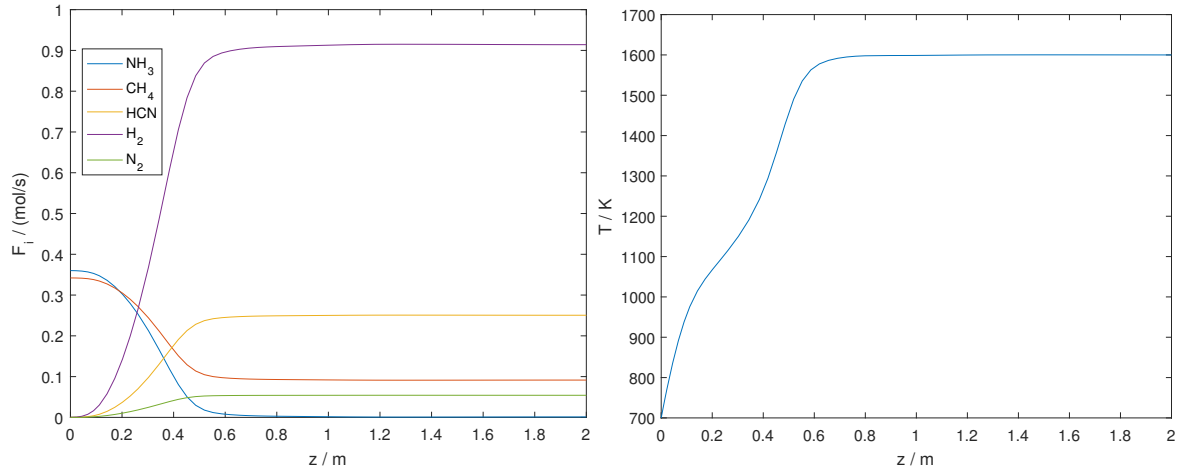

Figure 16: Flow rate- (*left*) and temperature (*right*) profile for the new obtained overall heat transfer coefficient and a very low NH_3 inlet flow rate of 0.36 mol/s are shown.

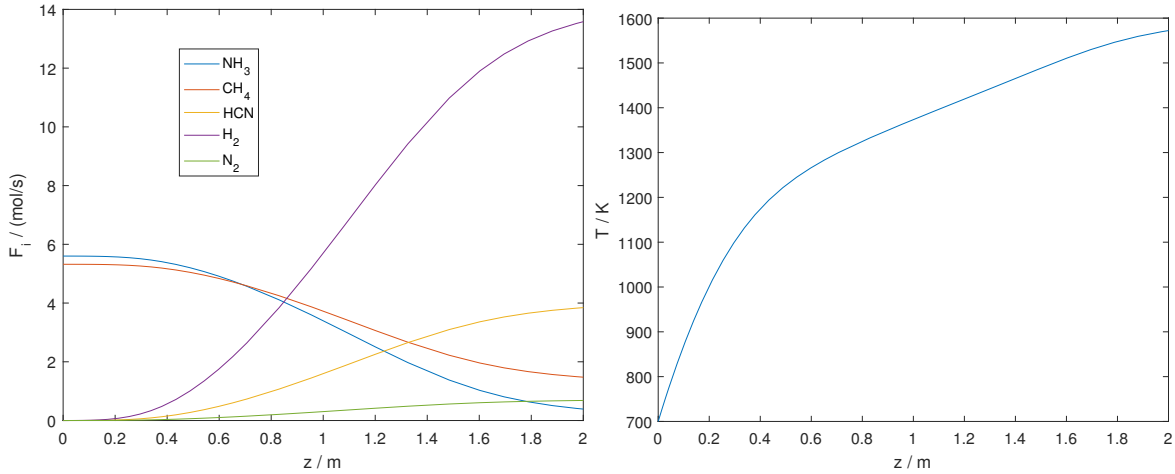

Figure 17: Optimized flux (*left*) and temperature (*right*) profile for the new obtained overall heat transfer coefficient and an NH_3 inlet flow rate of 5.6 mol/s are shown.

It can be seen, that the temperature cannot be increased fast enough to 1'600 K which delays the reaction start. Therefore, the production of HCN is reduced and more reaction tubes have to be present in order to produce the desired amount of HCN per year.

B Additional Data NH_3 Absorber Modelling

Table 9: Standard formation enthalpies at T=298 K and P=1 bar

Substance	$h_f^0[kJ/mol]$
NH_3	-45.94
CH_4	-74.60
HCN	135.14
H_2	0.0
N_2	0.0
H_2O	-735.13
H_2SO_4	-214.83
$(NH_4)_2SO_4$	-1181

Table 10: Shomate parameters for calculation of enthalpies in the NH_3 absorber

Substance	**A**	**B**	**C**	**D**
NH_3	19.996	49.771	-15.376	1.921
CH_4	-0.703	108.477	-42.552	5.863
HCN	32.694	22.592	-4.369	0.048
H_2	33.066	-11.363	11.433	-2.773
N_2	28.986	1.854	-9.647	16.635
H_2O	30.092	6.833	6.793	-2.534
H_2SO_4	47.289	190.331	-148.130	43.866

C Cost Calculation

Table 11: The expenses and revenues for the production plant over 10 years in USD.

Process part	Expenses	Revenues
Reactor System	$61.5 \cdot 10^6$	
Reactor device[a]	$0.6 \cdot 10^6$	
Maintenance[b]	$0.48 \cdot 10^6$	
Heat exchanger[c](preheat)	55'200	
Heat exchanger[c](cooler)	62'400	
Natural gas[17]	$5.92 \cdot 10^6$	
Cooling water	$4.56 \cdot 10^6$	
NH_3 feed[10]	$25.2 \cdot 10^6$	
CH_4 feed[10]	$24.5 \cdot 10^6$	
NH_3 Absorber (non-ideal)	$2.64 \cdot 10^6$	
Column	148'000	
Sulfuric acid	$2.31 \cdot 10^6$	
Cooling water	5'100	
Waste water	180'000	
HCN Absorber	210'650	$3.1 \cdot 10^7$
Total H_2 revenue		$3.1 \cdot 10^7$
Column	210'650	
Distillation (non-ideal)	$1.55 \cdot 10^6$	
TOTEX	$1.55 \cdot 10^6$	
Revenue of HCN		$147 \cdot 10^6$
Total profit of 10 years		$112 \cdot 10^6$

[a] One reactor block contains 400 pipes; [b] Replacement of pipes per year;
[c] Price depends on surface area;

D MATLAB Files Used for Calculations

```
%%
% Case Study II
% Reactor Design
% Authors: Ramona Achermann, Tim Forster
% Zurich, 28.4.16

function CaseStudy_ReactorDesignRealWithInnerHeatTransfer
clear
close all
clc

%=========================================================================
% Define Constants
%=========================================================================
n = 400;                            %amount of tubes
AmountReactor = 4;                  %amount of reactor blocks with 400 tubes
L = 2;                              %length of pipe in m
l = 2.5e-3;                         %Wall thickness in m
D = 15e-3;                          %Inner diameter of pipe in m
A_cross = (D / 2)^2 * pi;           %Cross sectional Area in m^2
a = 4 / D;                          %Surface/Volume ratio of a cylindrical pipe
lambda = 4.5;                       %thermal conductivity W / (m*s)
alpha = lambda / l;                 %thermal transition coefficient in W / (m^2 * s)
H = [251e3, 91e3];                  %reaction enthalpies for HCN production (1) and side reaction (2) in J/mol
Ta = 1600;                          %Heating temperature in K
P = 1*760;                          %initial pressure in torr
T0 = 700;                           %initial temperature in K
NA = 6.02e23;                       %Avogadro number mol^-1
workinghours = 8000 * 3600;         %working hours in a year

%=========================================================================
% Parameter for heat capacity
%=========================================================================
[Ap, Bp, Cp, Dp, Ep, Tcr] = ParameterHeatcapacity();

%=========================================================================
% Solve ODE to get slope of zero for the F-profile of HCN (low flux)
%=========================================================================
FNH3 = 0.0009;                          %mol/s
FCH4 = 0.95 * FNH3;                     %mol/s Ammonia 5% in excess
initial = [FNH3, FCH4, 0, 0, 0, T0];    %initial fluxes in mol/s
Vspan = [0, L * A_cross];               %integration range
[V, x] = ode15s(@(V, x) func(V, x, a, alpha, Ta, H, NA, Ap, Bp, Cp, Dp, Ep, Tcr, P, A_cross), Vspan, initial);
F = x(:, 1:5);                          %fluxes in mol/s
T = x(:, 6);                            %temperature in K
z = V ./ A_cross;                       %length of pipe reactor
yieldlow = F(end, 3) / F(1, 2);         %yield with this low flowrate
fprintf('Yield with low flow rate of NH3 (F=%g mol/s) is: yield=%.2g\n', FNH3, yieldlow)

%Plot the concentration profile
figure(1)
hold on
plot(z, F * n)
legend('NH_3', 'CH_4', 'HCN', 'H_2', 'N_2', 'location', 'best')
xlabel('z / m'), ylabel('F_i / (mol/s)')
ax = gca;
ax.YAxisLocation = 'left';
ax.Box = 'on';
% str=sprintf('F_{NH_3}=%g', FNH3);
% text(1.6, FNH3 * n, str,'FontWeight','bold')
saveas(figure(1),'PressureProfile_LowFlux_NonOptimized_withInnerHeattransfer','epsc');

%Plot the temperature profile
figure(2)
hold on
plot(z, T)
xlabel('z / m'), ylabel('T / K')
ax = gca;
ax.YAxisLocation = 'left';
ax.Box = 'on';
```

```
% str=sprintf('F_{NH_3}=%g', FNH3);
% text(1.6, 1350, str,'FontWeight','bold')
saveas(figure(2),'TemperatureProfile_LowFlux_NonOptimized_withInnerHeattransfer','epsc');

%=========================================================================
% Solve ODE with High flux
%=========================================================================
FNH3 = 0.07;                            %mol/s
FCH4 = 0.95 * FNH3;                     %mol/s Ammonia 5% in excess
initial = [FNH3, FCH4, 0, 0, 0, T0];    %initial fluxes in mol/s
Vspan = [0, L * A_cross];               %integration range
[V, x] = ode15s(@(V, x) func(V, x, a, alpha, Ta, H, NA, Ap, Bp, Cp, Dp, Ep, Tcr, P, A_cross), Vspan, initial);
F = x(:, 1:5);                          %fluxes in mol/s
T = x(:, 6);                            %temperature in K
z = V ./ A_cross;                       %length of pipe reactor
yieldhigh = F(end, 3) / F(1, 2);         %yield with this low flowrate
fprintf('Yield with high flow rate of NH3 (F=%g mol/s) is: yield=%.2g\n\n', FNH3, yieldhigh)

%Plot the concentration profile
figure(3)
hold on
plot(z, F * n)
legend('NH_3', 'CH_4', 'HCN', 'H_2', 'N_2', 'location', 'best')
xlabel('z / m'), ylabel('F_i / (mol/s)')
ax = gca;
ax.YAxisLocation = 'left';
ax.Box = 'on';
% str=sprintf('F_{NH_3} = %g', FNH3);
% text(1.6, FNH3 * n, str,'FontWeight','bold')
saveas(figure(3),'PressureProfile_HighFlux_NonOptimized_withInnerHeattransfer','epsc');

%Plot the temperature profile
figure(4)
hold on
plot(z, T)
xlabel('z / m'), ylabel('T / K')
ax = gca;
ax.YAxisLocation = 'left';
ax.Box = 'on';
% str=sprintf('F_{NH_3} = %g', FNH3);
% text(1.6, 1350, str,'FontWeight','bold')
saveas(figure(4),'TemperatureProfile_HighFlux_NonOptimized_withInnerHeattransfer','epsc');

%=========================================================================
% Optimise the flux FNH3 of the inlet
%=========================================================================
%Therefore, the yield (FHCN_out / FCH4_in) must be as large as possible (or
%vice versa: the inverse of this needs to be very small --> lsqnonlin
%minimizes this function!)

Fconv = 0.005;
guess = 0.015; %initial guess of the flux for NH3 0.009
options = optimset('display', 'off', 'MaxFunEvals', 100);
lowerbound = 0.014; %minimum 0.0483 mol/s per pipe for 1 reactor to get 10'000 t(HCN)/year
upperbound = 0.025;
% IF POSSIBILITY OF FCONV IS USED, SET FCONV ALSO IN THE FUNCTION FILE!
[optvar] = lsqnonlin(@(FNH3) optimisationfileyield(FNH3, L, A_cross, T0, a, alpha, Ta, H, NA, Ap, Bp, Cp, Dp,
    Ep, Tcr, P, Fconv), guess, lowerbound, upperbound, options);
fprintf('Optimised Flux for NH3 for ONE pipe (per reactor: multiply with 400): F = %.2g\n', optvar)
FNH3_opt = optvar;

%=========================================================================
% Draw optimised flux profile
%=========================================================================
FNH3 = FNH3_opt;                        %mol/s
FCH4 = 0.95 * FNH3;                     %mol/s Ammonia 5% in excess
initial = [FNH3, FCH4, 0, 0, 0, T0];    %initial fluxes in mol/s
Vspan = [0, L * A_cross];               %integration range
[V, x] = ode15s(@(V, x) func(V, x, a, alpha, Ta, H, NA, Ap, Bp, Cp, Dp, Ep, Tcr, P, A_cross), Vspan, initial);
F = x(:, 1:5);                          %fluxes in mol/s
T = x(:, 6);                            %temperature in K
z = V ./ A_cross;                       %length of pipe reactor
yieldopt = F(end, 3) / F(2, 1);       %yield for HCN in resp. to CH4
```

```
conversionopt = (F(1, 2) - F(end, 2) )/ F(1, 2);
fprintf('Optimised Conversion: y = %.2g\n', conversionopt)
fprintf('Optimised Yield: y = %.2g\n\n', yieldopt)
fprintf('Amount of reactors: %g\n', AmountReactor)
fprintf('Optimised outlet fluxes of ALL %g reactors (each with 400 tubes):\n\t FNH3 = %.3g mol/s\t    \n\t FCH4
      = %.3g mol/s\t     \n\t FHCN = %.3g mol/s\t     \n\t FH2 = %.3g mol/s\t    \n\t FN2 = %.3g mol/s\n',
     AmountReactor, F(end, :) .* AmountReactor .* n)

%Plot the concentration profile
figure(5)
hold on
plot(z, F * n)
legend('NH_3', 'CH_4', 'HCN', 'H_2', 'N_2', 'location', 'best')
xlabel('z / m'), ylabel('F_i / (mol/s)')
ax = gca;
ax.YAxisLocation = 'left';
ax.Box = 'on';
% str=sprintf('Optimized F_{NH_3} = %g', FNH3_opt);
% text(1.2, FNH3 * n, str,'FontWeight','bold')
saveas(figure(5),'PressureProfile_Optimized_withInnerHeattransfer','epsc');

%Plot the temperature profile
figure(6)
hold on
plot(z, T)
xlabel('z / m'), ylabel('T / K')
ax = gca;
ax.YAxisLocation = 'left';
ax.Box = 'on';
% str=sprintf('Optimized F_{NH_3} = %g', FNH3_opt);
% text(1.2, 1300,str,'FontWeight','bold')
saveas(figure(6),'TemperatureProfile_Optimized_withInnerHeattransfer','epsc');

% %========================================================================
% % Energy to Pre-heat
% %========================================================================
F_inPreheat = [FNH3_opt, FNH3_opt * 0.95] .* n .* AmountReactor;
% F_outReactor = [F(end, 1), F(end, 2), F(end, 3), F(end, 4), F(end, 5)] .* n .* AmountReactor;
% % [P_preheat, A_preheat, Cost_preheater] = EnergyPreheat(F_inPreheat);
% % fprintf('\nEnergy for preheaters needed per second: P=%.3g kW\n', P_preheat / 1000 )
% % fprintf('Surface of preheaters: A=%.3g m^2\n', A_preheat)
% % fprintf('Cost of all preheaters (device): A=%.3g m^2\n', Cost_preheater)
% [P_preheat, A_preheater, Cost_preheater] = HeatExchanger(F_inPreheat, F_outReactor);

%========================================================================
% Energy for Reactor
%========================================================================
F_outReactor = [F(end, 1), F(end, 2), F(end, 3), F(end, 4), F(end, 5)] .* n .* AmountReactor;
F_inReactor = [FNH3_opt, FNH3_opt * 0.95, 0, 0, 0] .* n .* AmountReactor;     %from optimisation
[P_reactor, mflux_naturalgas, cost_naturalgas] = EnergyReactor(F_outReactor, F_inReactor, H);
fprintf('\nEnergy for %g reactors per second: P=%.3g kW\n', AmountReactor, P_reactor / 1000)
fprintf('Amount of natural gas needed per second: m=%.3g kg/s\n', mflux_naturalgas)
fprintf('Cost of natural gas needed per year to heat %g reactors: USD=%.3g $\n', AmountReactor, cost_naturalgas
      * workinghours)

%========================================================================
% Cost for Reactor (device)
%========================================================================
cost_reactor =  AmountReactor * 150e3 + ... %US$ for ONE reactor with 400 tubes
                0.3 * n * AmountReactor * 100 * 10; %US$ for each tube to be replaced (30% replacement per year
                      times 10 years!)
fprintf('\nCost for all reactors (devices) and tubes per year: %g $', cost_reactor)

%========================================================================
% Energy for Cooling the gas stream
%========================================================================
% F_outReactor = [F(end, 1), F(end, 2), F(end, 3), F(end, 4), F(end, 5)] .* n .* AmountReactor;
% [P_coolingOutletGas, A_CoolingOutletGas, Cost_CoolingHeatExchanger] = EnergyCoolingOutletGas(F_outReactor);
% fprintf('\nEnergy for cooling the outlet gas needed per second: P=%.3g kW\n', P_coolingOutletGas / 1000 )
% fprintf('Surface of cooler for outlet gas: A=%.3g m^2\n', A_CoolingOutletGas)
% fprintf('Cost for all cooling heat exchangers (device): %g $\n', Cost_CoolingHeatExchanger)

%========================================================================
```

```
% Cost for starting materials
%=========================================================================
cost_StartingMaterial = CostStartingmaterial(F_inPreheat);
fprintf('\nCost of starting material (NH3) needed per year: %.3g $\n', cost_StartingMaterial(1) * workinghours)
fprintf('\nCost of starting material (CH4) needed per year: %.3g $\n', cost_StartingMaterial(2) * workinghours)

%=========================================================================
% Cost for cooling water (for cooling the outlet stream of the reactor)
%=========================================================================
% cost_coolingWater = CostCoolingWater(P_coolingOutletGas);
% fprintf('\nCost for the cooling water to cool the outlet stream of the reactor per year: %.3g $\n',
    cost_coolingWater * workinghours)
[Pcool, cost_Coolingwater, A_Cooling, cost_coolingHE, A_Heat, cost_preheatHE] = HeatExchanger(F_inReactor,
    F_outReactor);
fprintf('\nEnergy needed to cool the rxn gas (after preheater) per second: %.3g kW\n', Pcool / 1000);
fprintf('Surface of preheater: %.3g m^2\n', A_Heat);
fprintf('Surface of Cooling heat exchanger: %.3g m^2\n', A_Cooling);
fprintf('Cost of cooling water per year: %.3g $\n', cost_Coolingwater * workinghours);
fprintf('Cost of Preheater: %.3g $\n', cost_preheatHE);
fprintf('Cost of cooling heat exchanger: %.3g $\n', cost_coolingHE);

%=========================================================================
% Total operating expenses (OPEX) for preheat-reactor-cooling-system
%=========================================================================
OPEX =  cost_Coolingwater + ...             %cooling water US$/s
        sum(cost_StartingMaterial) + ...     %startmat US$/s
        cost_naturalgas;                    %natural gas US$/s
OPEX = OPEX * workinghours * 10;              %10 years total OPEX in US$
fprintf('\nOPEX (10 years): %.3g $\n', OPEX);

%=========================================================================
% Total capital expenses (CAPEX) for preheat-reactor-cooling-system
%=========================================================================
CAPEX = cost_reactor + ...      %reactor cost for 10 years in US$
        cost_coolingHE + ...    %heat exchanger (preheater) in US$
        cost_preheatHE;         %heat exchanger (cooler) in US$
fprintf('CAPEX (10 years): %.3g $\n', CAPEX);

%=========================================================================
% Total expenses (TOTEX) for preheat-reactor-cooling-system
%=========================================================================
TOTEX = CAPEX + OPEX;
fprintf('TOTEX (10 years): %.3g $\n', TOTEX);

end

%%
function dx = func(~, x, a, alpha, Ta, H, NA, Ap, Bp, Cp, Dp, Ep, Tcr, P0, A_cross)
% Function that defines the ODEs and calculates the heat capacities at
% every temperature in the reactor
% The variables are:
    % x(1)          ...NH3
    % x(2)          ...CH4
    % x(3)          ...HCN
    % x(4)          ...H2
    % x(5)          ...N2
    % x(6)          ...T

% Molar heat capacities of each species at each point/temperature in reactor
    cP = zeros();
    for i = 1:5 % Do it for all 5 species
        t = x(6) / 1000;
        if x(6) < Tcr(i)
            j = 1;
        else
            j = 2;
        end
        cP(i) = Ap(i,j) + Bp(i,j)*t + Cp(i,j)*t^2 + Dp(i,j)*t^3 + Ep(i,j)/t^2;
    end

    cP_CH4 = cP(1);
    cP_NH3 = cP(2);
    cP_HCN = cP(3);
```

```
    cP_H2 = cP(4);
    cP_N2 = cP(5);

%Total flux is defined as:
    Ftot = x(1) + x(2) + x(3) + x(4) + x(5);

%Calculate parameter for PR equation
    T = x(6);
    molefrac = x ./ Ftot;
    [Amix, Bmix, ~, ~, ~] = PR_Parameterfunc(T, molefrac, P0);

%Solve PR EoS
    phi = PR_EoS(Amix, Bmix);
    P = phi * P0;

%Calculate lambda for the certain temperature
    lambda = InterpolationLambda(x(6));
    lambdamix_rxn = sum((x(1:5) ./ Ftot) .* lambda(1:5));

%Calculate viscosity (mu) for the certain temperature
    mu = InterpolationDynamicViscositiy(T);
    mumix_rxn = sum((x(1:5) ./ Ftot) .* mu(1:5));

%Calculate viscosity (nu) for the certain temperature
    nu = InterpolationKinematicViscosity(T, P);
    numix_rxn = sum((x(1:5) ./ Ftot) .* nu(1:5));

%Calculate the gas velocity in the pipe (ideal)
    R = 0.062363577;                          %(m^3*torr)/(mol K)
    u_ideal = (Ftot * R * x(6)) / (P * A_cross);

%Calculate the gas velocity in the pipe over correction of the molar
%volumes by PR equation
    molarvolume = Molarvolume(T, P, x, Ftot, R);
    Q_real = sum(molarvolume) * Ftot;
    u_real = Q_real / A_cross;
disp({u_ideal, u_real, phi})
%Calculate Reynolds number for the gas in the tube (rxn-gas)
    D = 15e-3;
    Re_rxn = (u_real * D) / numix_rxn;

%Calculate Prandtl number for the gas in the tube (rxn-gas)
%     cP_mix = sum((x(1) / Ftot) * cP_NH3 + (x(2) / Ftot) * cP_CH4 + ...
%             (x(3) / Ftot) * cP_HCN + (x(4) / Ftot) * cP_H2 + (x(5) / Ftot) * cP_N2);
%     MW = [0.017, 0.016, 0.027, 0.002, 0.028];
%     MWmix = sum(molefrac(1:5)' .* MW);
%     Pr_rxn= (cP_mix * mumix_rxn) / (lambdamix_rxn * MWmix);
     Pr_rxn = 0.79;  %ASSUMPTION FROM BOOK

%Calculate Nusselt number for the gas in the tube (rxn-gas) with two
%different approaches
%     Nu_rxn = 0.02 .* (Re_rxn .^ 0.8) .* (Pr_rxn .^ 0.43);
    mu_end = InterpolationDynamicViscositiy(T);
    mumix_rxn_end = sum((x(1:5) ./ Ftot) .* mu_end(1:5));
    Nu_rxn = 0.027 * (Re_rxn^0.8) * (Pr_rxn^0.333) * ((mumix_rxn / mumix_rxn_end)^0.14);
    alpha_rxn = (Nu_rxn .* lambdamix_rxn) / D;

%Calculate overall heat transfer coefficient
    U = 1 / ((1 / alpha) + (1 / alpha_rxn));

%The rate expressions are defined separately in functions:
    rHCN = rateHCN(x(1), x(2), x(6), Ftot, NA, P);
    rN2 = rateN2(x(1), x(2), x(6), Ftot, NA, P);

%The mass balances and energy balance are defined as:
    dx(1) = - rHCN * a - 2 * rN2 * a;
    dx(2) = - rHCN * a;
    dx(3) = rHCN * a;
    dx(4) = 3 * rHCN * a + 3 * rN2 * a;
    dx(5) = rN2 * a;
    dx(6) = (U * a * (Ta - x(6)) - (rHCN * a *  H(1) + rN2 * a * H(2))) / (x(1) * cP_NH3 + x(2) * cP_CH4 + x(3)
          * cP_HCN + x(4) * cP_H2 + x(5) * cP_N2);
```

```
%Sum these up and send them back to the main file:
    dx = [dx(1); dx(2); dx(3); dx(4); dx(5); dx(6)];

end

%%
function rHCN = rateHCN(x1, x2, x6, Ftot, NA, P)
% rate expression of the reaction that produces HCN
% ATTENTION: units are mol / (s * m^2)

    rHCN =  (1e4 / NA) * ((7.8e18 * exp(-1950 / x6) * ((x2 * P) / Ftot) * (((x1 * P) / Ftot) ^ 0.5)) / ...
            ((1 + 0.044 * exp(2390 / x6) * ((x2 * P) / Ftot) * (((x1 * P) / Ftot) ^-0.5))^4));

end

%%
function rN2 = rateN2(x1, x2, x6, Ftot, NA, P)
% rate expression of the reaction that produces N2
% ATTENTION: units are mol / (s * m^2)

    rN2 =   (1e4 / NA) * ((4.9e18 * exp(-2130 / x6) * ((x1 * P) / Ftot)) / ...
            ((1 + 0.044 * exp(2390 / x6) * ((x2 * P) / Ftot) * (((x1 * P) / Ftot) ^ -0.5))^3));

end

%%
function frac = optimisationfileyield(optimvar , L, A_cross, T0, a, alpha, Ta, H, NA, Ap, Bp, Cp, Dp, Ep, Tcr,
     P, ~)
%optimvar(1)    ...FNH3
%optimvar(2)    ...ratio of CH4/NH3

    FCH4 = 0.95 * optimvar;
    Vspan = [0, L * A_cross];
    initial = [optimvar, FCH4, 0, 0, 0, T0];
    [~, x] = ode15s(@(V, x) func(V, x, a, alpha, Ta, H, NA, Ap, Bp, Cp, Dp, Ep, Tcr, P, A_cross), Vspan,
         initial);
    F = x(:, 1:5);                          %fluxes in mol/s

%     frac = -(F(end, 3) / F(1, 2));                %minimize this fraction
%     frac = (((F(1, 2) - F(end, 2)) / F(1, 2)) - 1)^2;     %minimize the conversion
    frac = (1 / (F(end, 3) / F(1, 2)) - 1)^2;      %minimize this fraction
%     frac = Fconv - F(end, 3);                       %minimize this fraction

end

%%
function Cost_StartingMaterial = CostStartingmaterial(F)
%This function calculates the cost for the starting materials in US$/s

%   F           ...inlet flux (mol/s) FOR ALL REACTORS for NH3 and CH4 with 5% excess of ammonia

    MW = [0.017, 0.016];                %molar weights of the starting material (kg/mol)
    mflux_startmat = F .* MW;           %mass flux of the starting material (kg/s)
    rawcost = [0.23, 0.25];             %cost of starting material in US$/kg

    Cost_StartingMaterial = mflux_startmat .* rawcost; %cost for starting material streams in US$/s

end

%%
function [Pcool, cost_Coolingwater, A_Cooling, cost_coolingHE, A_Heat, cost_preheatHE] = HeatExchanger(Fin,
     Fout)
%This file should calculate the energy needed to preheat the feed stream
%for ALL REACTORS !
%Constant pressure -> dQ = dH

    %Fin(1)         ...NH3 inlet Stream (mol/s)
    %Fin(2)         ...CH4 inlet Stream (mol/s)
    %Fout(1:5)      ...Fluxes of the outlet of the reactor in (mol/s)
    %T(1)           ...inlet temperature before the preheater in K
    %T(2)           ...inlet temperature before the reactor (after
    %                   preheater)
    %T(3)           ...outlet temperature after the reactor
```

```
425     %T(4)           ...temperature after cooling system (rxn gas goes into
426     %                   first absorber)
427     %Tcalc          ...outlet temperature after the cooler (crossflow with
428     %                   preheater)

430 %Define Temperature vector
431     Tfeed = [298, 700, 1600, 333];

433 %Get parameter for heat capacity
434     [Ap, Bp, Cp, Dp, Ep, Tcr] = ParameterHeatcapacity();

436 %Calculate the heat capacities
437     cP = zeros();
438     for z = 1 : 4 %Do it for inlet and outlet temperature
439         for i = 1:5 % Do it for all 5 species
440             t = Tfeed(z) ./ 1000;
441             if Tfeed(z) < Tcr(i)
442                 j = 1;
443             else
444                 j = 2;
445             end
446             cP(i, z) = Ap(i,j) + Bp(i,j).*t + Cp(i,j).*(t.^2) + Dp(i,j).*(t.^3) + Ep(i,j)./(t.^2); %in first
                    column: inlet cP
447         end
448     end

450     cP_CH4 = cP(1, :);  %(1,1)=inlet preheater, (1,2)=inlet reactor, (1,3)=outlet reactor
451     cP_NH3 = cP(2, :);
452     cP_HCN = cP(3, :);
453     cP_H2 = cP(4, :);
454     cP_N2 = cP(5, :);

456 %Define cP as well as a function of the temperature
457     cp_NH3 = @(T) Ap(2,2) + Bp(2,2).*(T/1000) + Cp(2,2).*((T/1000).^2) + Dp(2,2).*((T/1000).^3) + Ep(2,2)./((T
            /1000).^2);
458     cp_CH4 = @(T) Ap(1,2) + Bp(1,2).*(T/1000) + Cp(1,2).*((T/1000).^2) + Dp(1,2).*((T/1000).^3) + Ep(1,2)./((T
            /1000).^2);
459     cp_HCN = @(T) Ap(3,2) + Bp(3,2).*(T/1000) + Cp(3,2).*((T/1000).^2) + Dp(3,2).*((T/1000).^3) + Ep(3,2)./((T
            /1000).^2);
460     cp_H2 = @(T) Ap(4,2) + Bp(4,2).*(T/1000) + Cp(4,2).*((T/1000).^2) + Dp(4,2).*((T/1000).^3) + Ep(4,2)./((T
            /1000).^2);
461     cp_N2 = @(T) Ap(5,2) + Bp(5,2).*(T/1000) + Cp(5,2).*((T/1000).^2) + Dp(5,2).*((T/1000).^3) + Ep(5,2)./((T
            /1000).^2);

463 %Calculate enthalpy needed to heat up the feed
464     Pheat = Fin(1) * (cP_NH3(1, 2) * Tfeed(2) - cP_NH3(1, 1) * Tfeed(1)) + ...
465             Fin(2) * (cP_CH4(1, 2) * Tfeed(2) - cP_CH4(1, 1) * Tfeed(1));

467 %Calculate enthalpy needed to cool down the outlet stream of the reactor
468 %(this enthalpy depends on T which is reached after the heat exchanger)
469     Pcool = @(T)    Fout(1) * (cp_NH3(T) * T - cP_NH3(1, 3) * Tfeed(3)) + ...
470                     Fout(2) * (cp_CH4(T) * T - cP_CH4(1, 3) * Tfeed(3)) + ...
471                     Fout(3) * (cp_HCN(T) * T - cP_HCN(1, 3) * Tfeed(3)) + ...
472                     Fout(4) * (cp_H2(T) * T - cP_H2(1, 3) * Tfeed(3)) + ...
473                     Fout(5) * (cp_N2(T) * T - cP_N2(1, 3) * Tfeed(3));

475 %Calculate outlet temperature (of the outlet flux of the reactor) after the
476 %heat exchanger
477     options = optimset('display', 'off');
478     Tcalc = fsolve(@(T) Pheat + Pcool(T), 1400, options);

480 %Calculate power needed to cool the raction gas down to 333 K. With this
481 %information one can calculate the mass flux of the cooling water that
482 %is needed to cool this rection gas to the temperature needed for the
483 %first absorber (333 K)
484     Pcool = Fout(1) * (cP_NH3(1, 4) * Tfeed(4) - cp_NH3(Tcalc) * Tcalc) + ...
485             Fout(2) * (cP_CH4(1, 4) * Tfeed(4) - cp_CH4(Tcalc) * Tcalc) + ...
486             Fout(3) * (cP_HCN(1, 4) * Tfeed(4) - cp_HCN(Tcalc) * Tcalc) + ...
487             Fout(4) * (cP_H2(1, 4) * Tfeed(4) - cp_H2(Tcalc) * Tcalc) + ...
488             Fout(5) * (cP_N2(1, 4) * Tfeed(4) - cp_N2(Tcalc) * Tcalc);

490 %Assume inlet and outlet temperature of the cooling water
491     Tw = [15, 20] + 273.15;     %[K]
```

```
493 %Calculate mass flux of the cooling water needed to cool the rxn gas stream
494 %to 333 K (assuming that over this temperature range (~5 K) the heat
495 %capacity of water is constant
496     cp_H2O = 4.182e3;   %J/(kg K)
497     mflux_Coolingwater = Pcool / (cp_H2O * (Tw(1) - Tw(2)));    %kg/s

499 %Calculate cost of the cooling water per second (later multiplied with
500 %workinghours!)
501     cost_Coolingwater = mflux_Coolingwater * (0.1 / 1000);      %in US$/s (0.1US$/ton water)

503 %Calculate heat exchanger (cooling system) surface and cost of this device
504 %(HE = Heat Exchanger)
505     A_Cooling = Pcool / (850 * (Tfeed(4) - Tcalc));
506     cost_coolingHE = 25e3 * (A_Cooling ^ 0.65); %25'000 US$ per m^2 of HE

508 %Calculate heat exchanger (preheater) surface and cost of this device
509 %(HE = Heat Exchanger)
510     A_Heat = Pheat / (850 * (Tfeed(2) - Tfeed(1)));
511     cost_preheatHE = 25e3 * (A_Heat ^ 0.65); %25'000 US$ per m^2 of HE

513 end

515 %%
516 function [P_reactor, mflux_naturalgas, cost_naturalgas] = EnergyReactor(Fout, Fin, rH)
517 %This file should calculate the energy needed to preheat the feed streams
518 %for ALL REACTORS !
519 %Constant pressure -> dQ = dH

521     %F(1)           ...NH3 Stream flux
522     %F(2)           ...CH4 Stream flux

524 %Define Temperature vector
525     T = [700, 1600]; %inlet and outlet temperature

527 %Get parameter for heat capacity
528     [Ap, Bp, Cp, Dp, Ep, Tcr] = ParameterHeatcapacity();

530 %Calculate the heat capacities
531     cP = zeros();
532     for z = 1 : 2 %Do it for inlet and outlet temperature
533         for i = 1:5 % Do it for all 5 species
534             t = T(z) ./ 1000;
535             if T(z) < Tcr(i)
536                 j = 1;
537             else
538                 j = 2;
539             end
540             cP(i, z) = Ap(i,j) + Bp(i,j).*t + Cp(i,j).*(t.^2) + Dp(i,j).*(t.^3) + Ep(i,j)./(t.^2); %in first
                    column: inlet cP
541         end
542     end

544     cP_CH4 = cP(1, :);  %(1,1)=inlet, (1,2)=outlet
545     cP_NH3 = cP(2, :);
546     cP_HCN = cP(3, :);
547     cP_H2 = cP(4, :);
548     cP_N2 = cP(5, :);

550 %Calculate enthalpy needed
551     H = Fout(1) * (cP_NH3(1, 2) * T(2)) - Fin(1) * (cP_NH3(1, 1) * T(1)) + ...
552         Fout(2) * (cP_CH4(1, 2) * T(2)) - Fin(2) * (cP_CH4(1, 1) * T(1)) + ...
553         Fout(3) * (cP_HCN(1, 2) * T(2)) - Fin(3) * (cP_HCN(1, 1) * T(1)) + ...
554         Fout(4) * (cP_H2(1, 2) * T(2)) - Fin(4) * (cP_H2(1, 1) * T(1)) + ...
555         Fout(5) * (cP_N2(1, 2) * T(2)) - Fin(5) * (cP_N2(1, 1) * T(1)) + ...
556         rH(1) + rH(2);

558     P_reactor = H / 0.4; %divided by the preheater-efficiency

560 %Natural gas needed for this
561     rH_comb = 50e6;                 %heat of combustion in J/kg natural gas
562     mflux_naturalgas = H / rH_comb; %mass flux of natural gas needed in kg/s
563     cost_naturalgas = 0.250 * mflux_naturalgas; %cost of natural gas in $/s
```

```
565 end
567 %%
568 function [Amix, Bmix, E, amix, bmix] = PR_Parameterfunc(T, x, P0)
569 % This function calculates the parameter for the peng robinson equation
570 % 1: NH3
571 % 2: CH4
572 % 3: HCN
573 % 4: H2
574 % 5: N2
576 % T: temperature in K
577 % x: mole fractions (dimensionless)
579     R = 0.062363577;                                          %(m^3*torr)/(mol K)
580     P = P0 ;                                                  %ideal pressure in torr
581     Tc = [132.25, -82.59, 183.5, -240.01, -146.96] + 273.15;  %critical temperatures in K
582     pc = [113.3, 45.99, 53.9, 12.96, 33.96] .* 750.06;               %critical pressures in torr
584     Tr = T ./ Tc;                                             %reduced temperatures (dimensionless)
585     s = [0.37464, 1.54226, -0.26992];
586     omega = [0.25 , 0.011, 0.4095, -0.216, 0.039];            %acentric factors (dimensionless)
587     omegamix =  x(1) .* omega(1) + x(2) .* omega(2) + x(3) .* omega(3) + ...
588                 x(4) .* omega(4) + x(5) .* omega(5);
590     S = s(1) + s(2) .* omegamix + s(3) .* omegamix .^ 2;
591     k = (1 + S .* (1 - sqrt(Tr))) .^ 2;
592     E = S .* sqrt(Tr ./ k);
594     a = (0.457235 .* (R ^ 2) .* (Tc .^ 2) .* k) ./ pc;
595     b = (0.07780 .* R .* Tc) ./ (pc);
597     amix = (x(1) .* sqrt(a(1)) + x(2) .* sqrt(a(2)) + ...
598             x(3) .* sqrt(a(3)) + x(4) .* sqrt(a(4)) + ...
599             x(5) .* sqrt(a(5))) .^ 2;
601     bmix =  x(1) .* b(1) + x(2) .* b(2) + x(3) .* b(3) + ...
602             x(4) .* b(4) + x(5) .* b(5);
604     Amix = (amix .* P) ./ ((R .* T) .^ 2);
605     Bmix = (bmix .* P) ./ (R .* T);
607 end
609 %%
610 function phi = PR_EoS(Amix, Bmix)
612     coef = [1, (-1 + Bmix), (Amix - 2 * Bmix - 3 * Bmix^2), (-Amix * Bmix + Bmix^2 + Bmix^3)];
613     Z = roots(coef);
614     Z = max(Z);
615     phi = exp(Z - 1 - (Amix  / (2 * sqrt(2) * Bmix)) * log( ((Z + Bmix * (1 + sqrt(2)))) / (Z + Bmix * (1 -
          sqrt(2)))) - log(Z - Bmix));
617 end
619 %%
620 function lambdaint = InterpolationLambda(T)
621 % This function calculates the temperature dependence of the thermal
622 % conductivity by a linear fit and extrapolation (this is a plausible
623 % method --> see source in report for validity)
625 %=========================================================================
626 % Data thermal conductivity
627 %=========================================================================
628 TH2 = [100, 200, 300, 400];                         %temperature in K
629 lambdaH2 = [68.6, 131.7, 186.9, 230.4] ./ 1000;     %in W/(m*K)
631 TNH3 = [300, 400, 500, 600];
632 lambdaNH3 = [24.4, 37.4, 51.6, 66.8] ./ 1000;
634 TN2 = [100, 200, 300, 400, 500, 600];
635 lambdaN2 = [9.8, 18.7, 26.0, 32.3, 38.3, 44.0] ./ 1000;
```

```
TCH4 = [200, 300, 400, 500, 600];
lambdaCH4 = [22.5, 34.1, 49.1, 66.5, 84.1] ./ 1000;

THCN = [-17.8, -6.7, 4.4, 15.6, 26.7, 37.8, 48.9] + 273.15;     %temperature in K
lambdaHCN = [23.97, 25.62, 26.86, 28.1, 29.75, 30.99, 32.6] ./ 1000;    %(cal/(s*cm*K)) * 10^6

TO2 = [100, 200, 300, 400, 500, 600];
lambdaO2 = [9.3, 18.4, 26.3, 33.7, 41.0, 48.1] ./ 1000;

TCO2 = [200, 300, 400, 500, 600];
lambdaCO2 = [9.6, 16.8, 25.1, 33.5, 41.6] ./ 1000;

%==========================================================================
% Plots
%==========================================================================
% figure(1)
% hold on
% plot(TH2, lambdaH2, 'bx')
%
% figure(2)
% hold on
% plot(TNH3, lambdaNH3, 'bx')
%
% figure(3)
% hold on
% plot(TN2, lambdaN2, 'bx')
%
% figure(4)
% hold on
% plot(TCH4, lambdaCH4, 'bx')
%
% figure(5)
% hold on
% plot(THCN, lambdaHCN, 'bx')
%
% figure(6)
% hold on
% plot(TO2, lambdaO2, 'bx')
%
% figure(7)
% hold on
% plot(TCO2, lambdaCO2, 'bx')

%==========================================================================
% Linear fit
%==========================================================================
mdlH2 = fitlm(TH2, lambdaH2);
bH2 = mdlH2.Coefficients{1,1};
mH2 = mdlH2.Coefficients{2,1};
PolyH2 = @(T) mH2 * T + bH2;
lambdaintH2 = PolyH2(T);

mdlNH3 = fitlm(TNH3, lambdaNH3);
bNH3 = mdlNH3.Coefficients{1,1};
mNH3 = mdlNH3.Coefficients{2,1};
PolyNH3 = @(T) mNH3 * T + bNH3;
lambdaintNH3 = PolyNH3(T);

mdlN2 = fitlm(TN2, lambdaN2);
bN2 = mdlN2.Coefficients{1,1};
mN2 = mdlN2.Coefficients{2,1};
PolyN2 = @(T) mN2 * T + bN2;
lambdaintN2 = PolyN2(T);

mdlCH4 = fitlm(TCH4, lambdaCH4);
bCH4 = mdlCH4.Coefficients{1,1};
mCH4 = mdlCH4.Coefficients{2,1};
PolyCH4 = @(T) mCH4 * T + bCH4;
lambdaintCH4 = PolyCH4(T);

mdlHCN = fitlm(THCN, lambdaHCN);
bHCN = mdlHCN.Coefficients{1,1};
```

```
mHCN = mdlHCN.Coefficients{2,1};
PolyHCN = @(T) mHCN * T + bHCN;
lambdaintHCN = PolyHCN(T);

mdlO2 = fitlm(TO2, lambdaO2);
bO2 = mdlO2.Coefficients{1,1};
mO2 = mdlO2.Coefficients{2,1};
PolyO2 = @(T) mO2 * T + bO2;
lambdaintO2 = PolyO2(T);

mdlCO2 = fitlm(TCO2, lambdaCO2);
bCO2 = mdlCO2.Coefficients{1,1};
mCO2 = mdlCO2.Coefficients{2,1};
PolyCO2 = @(T) mCO2 * T + bCO2;
lambdaintCO2 = PolyCO2(T);

lambdaint = [lambdaintNH3, lambdaintCH4, lambdaintHCN, lambdaintH2, lambdaintN2, lambdaintO2, lambdaintCO2]';

%==========================================================================
% Plot Linear fit
%==========================================================================
% figure(1)
% plot(TH2, PolyH2(TH2), 'r-')
%
% figure(2)
% plot(TNH3, PolyNH3(TNH3), 'r-')
%
% figure(3)
% plot(TN2, PolyN2(TN2), 'r-')
%
% figure(4)
% plot(TCH4, PolyCH4(TCH4), 'r-')
%
% figure(5)
% plot(THCN, PolyHCN(THCN), 'r-')
%
% figure(6)
% plot(TO2, PolyO2(TO2), 'r-')
%
% figure(7)
% plot(TCO2, PolyCO2(TCO2), 'r-')

end

%%
function nu = InterpolationKinematicViscosity(T, P)
% This function calculates the temperature dependenc of the viscosity by
% using the southerland equatoin

    R = 0.062363577;                                        %(m^3*torr)/(mol K)
    MW = [0.017, 0.016, 0.027, 0.02, 0.028, 0.032, 0.044];  %kg / mol
    MWrxn = sum(MW(1:5)) / 5;                               %average MW for the rxn
    MWcomb = sum(MW(5:7))/2;
%     mu0 = [3.19, 2.56, 21.1, 1.84, 3.78, 4.47, 3.74] .* 1e-5;   %in Pa*s
%     T0 = [600, 600, 20, 600, 600, 600, 600] + 273.15;           %in K
    rhorxn = (P * MWrxn) / (R * T);
    rhocomb =(P * MWcomb) / (R * T);

%     C = [370, 169, -, 72, 111, 127, 240];                       %Southerland constants in K
%     mu = mu0 .* ((T0 + C)./ (T + C)) .* (T / T0) .^ (3 / 2);    %Southerland formel

    mu = InterpolationDynamicViscositiy(T);

    nurxn = mu(1:5) ./ rhorxn;
    nucomb = mu(6:7) ./ rhocomb;
    nu = [nurxn; nucomb];

end

%%
function muint = InterpolationDynamicViscositiy(T)
%==========================================================================
% Data mu
```

```
%=========================================================================
TH2 = [0, 20, 50, 100, 200, 300, 400, 500, 600] + 273.15;               %temperature in K
nuH2 = [0.84, 0.88, 0.94, 1.04, 1.21, 1.37, 1.53, 1.69, 1.84] .* 1e-5;  %in W/(m*K)

TNH3 = [0, 20, 50, 100, 200, 300, 400, 500, 600] + 273.15;              %K
nuNH3 = [0.92, 0.99, 1.1, 1.3, 1.68, 2.06, 2.44, 2.82, 3.19] .* 1e-5;   %Pa*s

TN2 = [0, 20, 50, 100, 200, 300, 400, 500, 600] + 273.15;
nuN2 = [1.66, 1.76, 1.89, 2.12, 2.51, 2.86, 3.19, 3.49, 3.78] .* 1e-5;

TCH4 = [0, 20, 50, 100, 200, 300, 400, 500, 600] + 273.15;
nuCH4 = [1.03, 1.1, 1.19, 1.35, 1.63, 1.88, 2.11, 2.33, 2.53] .* 1e-5;

%NOT CORRECT!
THCN = [0, 20, 50, 100, 200, 300, 400, 500, 600] + 273.15;     %temperature in K
nuHCN = [1.37, 1.47, 1.61, 1.84, 2.29, 2.7, 3.07, 3.4, 3.7] .* 1e-5;    %(cal/(s*cm*K)) * 10^6

TO2 = [0, 20, 50, 100, 200, 300, 400, 500, 600] + 273.15;
nuO2 = [1.95, 2.04, 2.18, 2.44, 2.93, 3.37, 3.76, 4.13, 4.47] .* 1e-5;

TCO2 = [0, 20, 50, 100, 200, 300, 400, 500, 600] + 273.15;
nuCO2 = [1.37, 1.47, 1.61, 1.85, 2.3, 2.71, 3.08, 3.42, 3.74] .* 1e-5;

%=========================================================================
% Plots
%=========================================================================
% figure(1)
% hold on
% plot(TH2, nuH2, 'bx')
%
% figure(2)
% hold on
% plot(TNH3, nuNH3, 'bx')
%
% figure(3)
% hold on
% plot(TN2, nuN2, 'bx')
%
% figure(4)
% hold on
% plot(TCH4, nuCH4, 'bx')
%
% figure(5)
% hold on
% plot(THCN, nuHCN, 'bx')
%
% figure(6)
% hold on
% plot(TO2, nuO2, 'bx')
%
% figure(7)
% hold on
% plot(TCO2, nuCO2, 'bx')

%=========================================================================
% Linear fit
%=========================================================================
mdlH2 = polyfit(TH2, nuH2, 2);
bH2 = mdlH2(1);
mH2 = mdlH2(2);
uH2 = mdlH2(3);
PolyH2 = @(T) bH2 .* (T .^2) + mH2 .* T + uH2;
nuintH2 = PolyH2(T);

mdlNH3 = polyfit(TNH3, nuNH3, 2);
bNH3 = mdlNH3(1);
mNH3 = mdlNH3(2);
uNH3 = mdlNH3(3);
PolyNH3 = @(T) bNH3 .* (T .^2) + mNH3 .* T + uNH3;
nuintNH3 = PolyNH3(T);

mdlCH4 = polyfit(TCH4, nuCH4, 2);
bCH4 = mdlCH4(1);
```

```
mCH4 = mdlCH4(2);
uCH4 = mdlCH4(3);
PolyCH4 = @(T) bCH4 .* (T .^2) + mCH4 .* T + uCH4;
nuintCH4 = PolyCH4(T);

mdlHCN = polyfit(THCN, nuHCN, 2);
bHCN = mdlHCN(1);
mHCN = mdlHCN(2);
uHCN = mdlHCN(3);
PolyHCN = @(T) bHCN .* (T .^2) + mHCN .* T + uHCN;
nuintHCN = PolyHCN(T);

mdlN2 = polyfit(TN2, nuN2, 2);
bN2 = mdlN2(1);
mN2 = mdlN2(2);
uN2 = mdlN2(3);
PolyN2 = @(T) bN2 .* (T .^2) + mN2 .* T + uN2;
nuintN2 = PolyN2(T);

mdlO2 = polyfit(TO2, nuO2, 2);
bO2 = mdlO2(1);
mO2 = mdlO2(2);
uO2 = mdlO2(3);
PolyO2 = @(T) bO2 .* (T .^2) + mO2 .* T + uO2;
nuintO2 = PolyO2(T);

mdlCO2 = polyfit(TCO2, nuCO2, 2);
bCO2 = mdlCO2(1);
mCO2 = mdlCO2(2);
uCO2 = mdlCO2(3);
PolyCO2 = @(T) bCO2 .* (T .^2) + mCO2 .* T + uCO2;
nuintCO2 = PolyCO2(T);

muint = [nuintNH3, nuintCH4, nuintHCN, nuintH2, nuintN2, nuintO2, nuintCO2]';

%=========================================================================
% Plot Linear fit
%=========================================================================
% figure(1)
% plot(TH2, PolyH2(TH2), 'r-')
%
% figure(2)
% plot(TNH3, PolyNH3(TNH3), 'r-')
%
% figure(3)
% plot(TN2, PolyN2(TN2), 'r-')
%
% figure(4)
% plot(TCH4, PolyCH4(TCH4), 'r-')
%
% figure(5)
% plot(THCN, PolyHCN(THCN), 'r-')
%
% figure(6)
% plot(TO2, PolyO2(TO2), 'r-')
%
% figure(7)
% plot(TCO2, PolyCO2(TCO2), 'r-')

end

%%
function [Ap, Bp, Cp, Dp, Ep, Tcr] = ParameterHeatcapacity()
% CH4
A_CH4 = [-0.703029,  85.81217];
B_CH4 = [108.4773,  11.26467];
C_CH4 = [-42.52157,  -2.114146];
D_CH4 = [5.862788,  0.138190];
E_CH4 = [0.678565,  -26.42221];
T_CH4 = 1300; % [K] temperature, where the second parameter starts

% NH3
A_NH3 = [19.99563  ,52.02427];
```

```
B_NH3 = [49.77119,  18.48801];
C_NH3 = [-15.37599,  -3.765128];
D_NH3 = [1.921168  ,0.248541];
E_NH3 = [0.189174,  -12.45799];
T_NH3 = 1400;

% HCN
A_HCN = [32.69373  ,52.36527];
B_HCN = [22.59205,  5.563298];
C_HCN = [-4.369142,  -0.953224];
D_HCN = [-0.407697,  0.056711];
E_HCN = [-0.282399,  -7.564086];
T_HCN = 1300;

% H2
A_H2 = [33.066178,  18.563083];
B_H2 = [-11.363417,  12.257357];
C_H2 = [11.432816  ,-2.859786];
D_H2 = [-2.772874,  0.268238];
E_H2 = [-0.158558  ,1.977990];
T_H2 = 1000;

% N2 - still wrong!!
A_N2 = [19.50583,  19.50583];
B_N2 = [19.88705,  19.88705];
C_N2 = [-8.598535, -8.598535];
D_N2 = [1.369784,  1.369784];
E_N2 = [0.527601  , 0.527601];
T_N2 = 1000;

% Summariezed parameter for ODE
Ap = [A_CH4;A_NH3;A_HCN;A_H2;A_N2];
Bp = [B_CH4;B_NH3;B_HCN;B_H2;B_N2];
Cp = [C_CH4;C_NH3;C_HCN;C_H2;C_N2];
Dp = [D_CH4;D_NH3;D_HCN;D_H2;D_N2];
Ep = [E_CH4;E_NH3;E_HCN;E_H2;E_N2];
Tcr = [T_CH4,T_NH3,T_HCN,T_H2,T_N2];
end

%%
function molarvolume = Molarvolume(T, P, F, Ftot, R)

%     molefrac = F(1:5) ./ Ftot;                                        %molefraction (dimensionless)
%     [~, ~, ~, a, b] = PR_Parameterfunc(T, molefrac, P);     %parameter from PR equation
%     guess = (R * T) / P;                                      %guessed initial value for the molar volume at a
     temperature T
%
%     options = optimset('display', 'off');
%     vNH3 = lsqnonlin(@(v) (((R * T) / (v - b(1))) - (a(1) / (v^2 + 2 * v * b(1) - b(1)^2))) - P * molefrac(1)
     , guess, [], [], options);
%     vCH4 = lsqnonlin(@(v) (((R * T) / (v - b(2))) - (a(2) / (v^2 + 2 * v * b(2) - b(2)^2))) - P * molefrac(2)
     , guess, [], [], options);
%     vHCN = lsqnonlin(@(v) (((R * T) / (v - b(3))) - (a(3) / (v^2 + 2 * v * b(3) - b(3)^2))) - P * molefrac(3)
     , guess, [], [], options);
%     vH2 = lsqnonlin(@(v) (((R * T) / (v - b(4))) - (a(4) / (v^2 + 2 * v * b(4) - b(4)^2))) - P * molefrac(4),
      guess, [], [],options);
%     vN2 = lsqnonlin(@(v) (((R * T) / (v - b(5))) - (a(5) / (v^2 + 2 * v * b(5) - b(5)^2))) - P * molefrac(5),
      guess, [], [],options);
%
%     molarvolume = [vNH3; vCH4; vHCN; vH2; vN2];

 molefrac = F(1:5) ./ Ftot;                                        %molefraction (dimensionless)
    [~, ~, ~, a, b] = PR_Parameterfunc(T, molefrac, P);     %parameter from PR equation
    guess = (R * T) / P;                                      %guessed initial value for the molar volume at a
         temperature T

    options = optimset('display', 'off');
    Vm = lsqnonlin(@(v) (((R * T) / (v - b)) - (a / (v^2 + 2 * v * b - b^2))) - P, guess, [], [], options);

    vNH3 = Vm * molefrac(1);
    vCH4 = Vm * molefrac(2);
    vHCN = Vm * molefrac(3);
    vH2 = Vm * molefrac(4);
```

```
    vN2 = Vm * molefrac(5);

    molarvolume = [vNH3; vCH4; vHCN; vH2; vN2];
end
```

Listing 1: File used in section 3.1 to model the reactor system considering temperature dependencies and non ideality.

```
%%
% Case Study II
% Reactor Design
% Authors: Ramona Achermann, Tim Forster
% Zurich, 28.4.16

function CaseStudy_ReactorDesignPressureDependence
clear
close all
clc

%=========================================================================
% Define Constants
%=========================================================================
n = 400;                            %amount of tubes
L = 2;                              %length of pipe in m
l = 2.5e-3;                         %Wall thickness in m
D = 15e-3;                          %Inner diameter of pipe in m
A_cross = (D / 2)^2 * pi;           %Cross sectional Area in m^2
a = 4 / D;                          %Surface/Volume ratio of a cylindrical pipe
lambda = 4.5;                       %thermal conductivity W / (m*s)
alpha = lambda / l;                 %thermal transition coefficient in W / (m^2 * s)
H = [251e3, 91e3];                  %reaction enthalpies for HCN production (1) and side reaction (2) in J/mol
Ta = 1600;                          %Heating temperature in K
P = [1, 5, 10, 25, 50, 100] .* 760; %initial pressure in torr
T0 = 700;                           %initial temperature in K
NA = 6.02e23;                       %Avogadro number mol^-1

%=========================================================================
% Parameter for heat capacity
%=========================================================================
[Ap, Bp, Cp, Dp, Ep, Tcr] = ParameterHeatcapacity();

%=========================================================================
% Solve ODE
%=========================================================================
yield=zeros();
for i = 1 : 6

    FNH3 = 0.009;                           %mol/s
    FCH4 = 0.95 * FNH3;                     %mol/s Ammonia 5% in excess
    initial = [FNH3, FCH4, 0, 0, 0, T0];    %initial fluxes in mol/s
    Vspan = [0, L * A_cross];               %integration range
    [V, x] = ode15s(@(V, x) func(V, x, a, alpha, Ta, H, NA, Ap, Bp, Cp, Dp, Ep, Tcr, P(i), A_cross), Vspan,
         initial);
    F = x(:, 1:5);                          %fluxes in mol/s
    z = V ./ A_cross;                       %length of pipe reactor

    %Plot the concentration profile
    figure(1)
    hold on
    plot(z, F(:, 3) .* n)
    yield(i) = F(end, 3) / F(1, 2);

end

legend(sprintf('P=%g atm', P(1)/760), sprintf('P=%g atm', P(2)/760), sprintf('P=%g atm', P(3)/760), sprintf('P
     =%g atm', P(4)/760), sprintf('P=%g atm', P(5)/760), sprintf('P=%g atm', P(6)/760), 'location', 'best')
xlabel('z / m'), ylabel('F_{HCN} / (mol/s)')
ax = gca;
ax.YAxisLocation = 'left';
ax.Box = 'on';
saveas(figure(1),'PressureDependenceHCNCurve','epsc');
```

```
fprintf('Yield with P=%g atm: %g\n', P(1)/760, yield(1)*100)
fprintf('Yield with P=%g atm: %g\n', P(2)/760, yield(2)*100)
fprintf('Yield with P=%g atm: %g\n', P(3)/760, yield(3)*100)
fprintf('Yield with P=%g atm: %g\n', P(4)/760, yield(4)*100)
fprintf('Yield with P=%g atm: %g\n', P(5)/760, yield(5)*100)
fprintf('Yield with P=%g atm: %g\n', P(6)/760, yield(6)*100)

%more pressures
P = [1, 2, 3, 4, 5, 7, 10, 15, 20, 25, 30, 40, 50, 60, 70, 80, 90, 100] .* 760; %initial pressure in torr
for i = 1 : length(P)

    FNH3 = 0.009;                           %mol/s
    FCH4 = 0.95 * FNH3;                     %mol/s Ammonia 5% in excess
    initial = [FNH3, FCH4, 0, 0, 0, T0];    %initial fluxes in mol/s
    Vspan = [0, L * A_cross];               %integration range
    [~, x] = ode15s(@(V, x) func(V, x, a, alpha, Ta, H, NA, Ap, Bp, Cp, Dp, Ep, Tcr, P(i), A_cross), Vspan,
         initial);
    F = x(:, 1:5);                          %fluxes in mol/s
    yield(i) = F(end, 3) / F(1, 2);

end

figure(2)
hold on
plot(P./760, yield, 'r.-')
xlabel('Pressure / atm'), ylabel('Yield / [-]')
ax = gca;
ax.YAxisLocation = 'left';
ax.Box = 'on';
saveas(figure(2),'PressureDependenceYield','epsc');

end

%%
function dx = func(~, x, a, alpha, Ta, H, NA, Ap, Bp, Cp, Dp, Ep, Tcr, P0, A_cross)
% Function that defines the ODEs and calculates the heat capacities at
% every temperature in the reactor
% The variables are:
    % x(1)          ...NH3
    % x(2)          ...CH4
    % x(3)          ...HCN
    % x(4)          ...H2
    % x(5)          ...N2
    % x(6)          ...T

% Molar heat capacities of each species at each point/temperature in reactor
    cP = zeros();
    for i = 1:5 % Do it for all 5 species
        t = x(6) / 1000;
        if x(6) < Tcr(i)
            j = 1;
        else
            j = 2;
        end
        cP(i) = Ap(i,j) + Bp(i,j)*t + Cp(i,j)*t^2 + Dp(i,j)*t^3 + Ep(i,j)/t^2;
    end

    cP_CH4 = cP(1);
    cP_NH3 = cP(2);
    cP_HCN = cP(3);
    cP_H2 = cP(4);
    cP_N2 = cP(5);

%Total flux is defined as:
    Ftot = x(1) + x(2) + x(3) + x(4) + x(5);

%Calculate parameter for PR equation
    T = x(6);
    molefrac = x ./ Ftot;
    [Amix, Bmix, ~, ~, ~] = PR_Parameterfunc(T, molefrac, P0);

%Solve PR EoS
    phi = PR_EoS(Amix, Bmix);
```

```
    P = phi * P0;

%Calculate lambda for the certain temperature
    lambda = InterpolationLambda(x(6));
    lambdamix_rxn = sum((x(1:5) ./ Ftot) .* lambda(1:5));

%Calculate viscosity (mu) for the certain temperature
    mu = InterpolationDynamicViscositiy(T);
    mumix_rxn = sum((x(1:5) ./ Ftot) .* mu(1:5));

%Calculate viscosity (nu) for the certain temperature
    nu = InterpolationKinematicViscosity(T, P);
    numix_rxn = sum((x(1:5) ./ Ftot) .* nu(1:5));

%Calculate the gas velocity in the pipe (ideal)
    R = 0.062363577;                          %(m^3*torr)/(mol K)
%     u_ideal = (Ftot * R * x(6)) / (P * A_cross);

%Calculate the gas velocity in the pipe over correction of the molar
%volumes by PR equation
    molarvolume = Molarvolume(T, P, x, Ftot, R);
    Q_real = sum(molarvolume) * Ftot;
    u_real = Q_real / A_cross;

%Calculate Reynolds number for the gas in the tube (rxn-gas)
    D = 15e-3;
    Re_rxn = (u_real * D) / numix_rxn;

%Calculate Prandtl number for the gas in the tube (rxn-gas)
%     cP_mix = sum((x(1) / Ftot) * cP_NH3 + (x(2) / Ftot) * cP_CH4 + ...
%             (x(3) / Ftot) * cP_HCN + (x(4) / Ftot) * cP_H2 + (x(5) / Ftot) * cP_N2);
%     Pr_rxn= (cP_mix * mumix_rxn) / lambdamix_rxn;
     Pr_rxn = 0.79;  %ASSUMPTION FROM BOOK

%Calculate Nusselt number for the gas in the tube (rxn-gas) with two
%different approaches
%     Nu_rxn = 0.02 .* (Re_rxn .^ 0.8) .* (Pr_rxn .^ 0.43);
    mu_end = InterpolationDynamicViscositiy(T);
    mumix_rxn_end = sum((x(1:5) ./ Ftot) .* mu_end(1:5));
    Nu_rxn = 0.027 * (Re_rxn^0.8) * (Pr_rxn^0.333) * ((mumix_rxn / mumix_rxn_end)^0.14);
    alpha_rxn = (Nu_rxn .* lambdamix_rxn) / D;

%Calculate overall heat transfer coefficient
    U = 1 / ((1 / alpha) + (1 / alpha_rxn));

%The rate expressions are defined separately in functions:
    rHCN = rateHCN(x(1), x(2), x(6), Ftot, NA, P);
    rN2 = rateN2(x(1), x(2), x(6), Ftot, NA, P);

%The mass balances and energy balance are defined as:
    dx(1) = - rHCN * a - 2 * rN2 * a;
    dx(2) = - rHCN * a;
    dx(3) = rHCN * a;
    dx(4) = 3 * rHCN * a + 3 * rN2 * a;
    dx(5) = rN2 * a;
    dx(6) = (U * a * (Ta - x(6)) - (rHCN * a *  H(1) + rN2 * a * H(2))) / (x(1) * cP_NH3 + x(2) * cP_CH4 + x(3)
          * cP_HCN + x(4) * cP_H2 + x(5) * cP_N2);

%Sum these up and send them back to the main file:
    dx = [dx(1); dx(2); dx(3); dx(4); dx(5); dx(6)];

end

%%
function rHCN = rateHCN(x1, x2, x6, Ftot, NA, P)
% rate expression of the reaction that produces HCN
% ATTENTION: units are mol / (s * m^2)

    rHCN =  (1e4 / NA) * ((7.8e18 * exp(-1950 / x6) * ((x2 * P) / Ftot) * (((x1 * P) / Ftot) ^ 0.5)) / ...
            ((1 + 0.044 * exp(2390 / x6) * ((x2 * P) / Ftot) * (((x1 * P) / Ftot) ^-0.5))^4));

end
```

```
%%
function rN2 = rateN2(x1, x2, x6, Ftot, NA, P)
% rate expression of the reaction that produces N2
% ATTENTION: units are mol / (s * m^2)

    rN2 =   (1e4 / NA) * ((4.9e18 * exp(-2130 / x6) * ((x1 * P) / Ftot)) / ...
            ((1 + 0.044 * exp(2390 / x6) * ((x2 * P) / Ftot) * (((x1 * P) / Ftot) ^ -0.5))^3));

end

%%
function [Amix, Bmix, E, amix, bmix] = PR_Parameterfunc(T, x, P0)
% This function calculates the parameter for the peng robinson equation
% 1: NH3
% 2: CH4
% 3: HCN
% 4: H2
% 5: N2

% T: temperature in K
% x: mole fractions (dimensionless)

    R = 0.062363577;                                            %(m^3*torr)/(mol K)
    P = P0 ;                                                    %ideal pressure in torr
    Tc = [132.25, -82.59, 183.5, -240.01, -146.96] + 273.15;    %critical temperatures in K
    pc = [113.3, 45.99, 53.9, 12.96, 33.96] .* 750.06;                 %critical pressures in torr

    Tr = T ./ Tc;                                               %reduced temperatures (dimensionless)
    s = [0.37464, 1.54226, -0.26992];
    omega = [0.25 , 0.011, 0.4095, -0.216, 0.039];              %acentric factors (dimensionless)
    omegamix =  x(1) .* omega(1) + x(2) .* omega(2) + x(3) .* omega(3) + ...
                x(4) .* omega(4) + x(5) .* omega(5);

    S = s(1) + s(2) .* omegamix + s(3) .* omegamix .^ 2;
    k = (1 + S .* (1 - sqrt(Tr))) .^ 2;
    E = S .* sqrt(Tr ./ k);

    a = (0.457235 .* (R ^ 2) .* (Tc .^ 2) .* k) ./ pc;
    b = (0.07780 .* R .* Tc) ./ (pc);

    amix = (x(1) .* sqrt(a(1)) + x(2) .* sqrt(a(2)) + ...
            x(3) .* sqrt(a(3)) + x(4) .* sqrt(a(4)) + ...
            x(5) .* sqrt(a(5))) .^ 2;

    bmix =  x(1) .* b(1) + x(2) .* b(2) + x(3) .* b(3) + ...
            x(4) .* b(4) + x(5) .* b(5);

    Amix = (amix .* P) ./ ((R .* T) .^ 2);
    Bmix = (bmix .* P) ./ (R .* T);

end

%%
function phi = PR_EoS(Amix, Bmix)

    coef = [1, (-1 + Bmix), (Amix - 2 * Bmix - 3 * Bmix^2), (-Amix * Bmix + Bmix^2 + Bmix^3)];
    Z = roots(coef);
    Z = max(Z);
    phi = exp(Z - 1 - (Amix  / (2 * sqrt(2) * Bmix)) * log( ((Z + Bmix * (1 + sqrt(2)))) / (Z + Bmix * (1 -
         sqrt(2)))) - log(Z - Bmix));

end

%%
function lambdaint = InterpolationLambda(T)
% This function calculates the temperature dependence of the thermal
% conductivity by a linear fit and extrapolation (this is a plausible
% method --> see source in report for validity)

%=========================================================================
% Data thermal conductivity
%=========================================================================
TH2 = [100, 200, 300, 400];                         %temperature in K
```

```
lambdaH2 = [68.6, 131.7, 186.9, 230.4] ./ 1000;     %in W/(m*K)

TNH3 = [300, 400, 500, 600];
lambdaNH3 = [24.4, 37.4, 51.6, 66.8] ./ 1000;

TN2 = [100, 200, 300, 400, 500, 600];
lambdaN2 = [9.8, 18.7, 26.0, 32.3, 38.3, 44.0] ./ 1000;

TCH4 = [200, 300, 400, 500, 600];
lambdaCH4 = [22.5, 34.1, 49.1, 66.5, 84.1] ./ 1000;

THCN = [-17.8, -6.7, 4.4, 15.6, 26.7, 37.8, 48.9] + 273.15;     %temperature in K
lambdaHCN = [23.97, 25.62, 26.86, 28.1, 29.75, 30.99, 32.6] ./ 1000;    %(cal/(s*cm*K)) * 10^6

TO2 = [100, 200, 300, 400, 500, 600];
lambdaO2 = [9.3, 18.4, 26.3, 33.7, 41.0, 48.1] ./ 1000;

TCO2 = [200, 300, 400, 500, 600];
lambdaCO2 = [9.6, 16.8, 25.1, 33.5, 41.6] ./ 1000;

%=========================================================================
% Plots
%=========================================================================
% figure(1)
% hold on
% plot(TH2, lambdaH2, 'bx')
%
% figure(2)
% hold on
% plot(TNH3, lambdaNH3, 'bx')
%
% figure(3)
% hold on
% plot(TN2, lambdaN2, 'bx')
%
% figure(4)
% hold on
% plot(TCH4, lambdaCH4, 'bx')
%
% figure(5)
% hold on
% plot(THCN, lambdaHCN, 'bx')
%
% figure(6)
% hold on
% plot(TO2, lambdaO2, 'bx')
%
% figure(7)
% hold on
% plot(TCO2, lambdaCO2, 'bx')

%=========================================================================
% Linear fit
%=========================================================================
mdlH2 = fitlm(TH2, lambdaH2);
bH2 = mdlH2.Coefficients{1,1};
mH2 = mdlH2.Coefficients{2,1};
PolyH2 = @(T) mH2 * T + bH2;
lambdaintH2 = PolyH2(T);

mdlNH3 = fitlm(TNH3, lambdaNH3);
bNH3 = mdlNH3.Coefficients{1,1};
mNH3 = mdlNH3.Coefficients{2,1};
PolyNH3 = @(T) mNH3 * T + bNH3;
lambdaintNH3 = PolyNH3(T);

mdlN2 = fitlm(TN2, lambdaN2);
bN2 = mdlN2.Coefficients{1,1};
mN2 = mdlN2.Coefficients{2,1};
PolyN2 = @(T) mN2 * T + bN2;
lambdaintN2 = PolyN2(T);

mdlCH4 = fitlm(TCH4, lambdaCH4);
```

```
bCH4 = mdlCH4.Coefficients{1,1};
mCH4 = mdlCH4.Coefficients{2,1};
PolyCH4 = @(T) mCH4 * T + bCH4;
lambdaintCH4 = PolyCH4(T);

mdlHCN = fitlm(THCN, lambdaHCN);
bHCN = mdlHCN.Coefficients{1,1};
mHCN = mdlHCN.Coefficients{2,1};
PolyHCN = @(T) mHCN * T + bHCN;
lambdaintHCN = PolyHCN(T);

mdlO2 = fitlm(TO2, lambdaO2);
bO2 = mdlO2.Coefficients{1,1};
mO2 = mdlO2.Coefficients{2,1};
PolyO2 = @(T) mO2 * T + bO2;
lambdaintO2 = PolyO2(T);

mdlCO2 = fitlm(TCO2, lambdaCO2);
bCO2 = mdlCO2.Coefficients{1,1};
mCO2 = mdlCO2.Coefficients{2,1};
PolyCO2 = @(T) mCO2 * T + bCO2;
lambdaintCO2 = PolyCO2(T);

lambdaint = [lambdaintNH3, lambdaintCH4, lambdaintHCN, lambdaintH2, lambdaintN2, lambdaintO2, lambdaintCO2]';

%=========================================================================
% Plot Linear fit
%=========================================================================
% figure(1)
% plot(TH2, PolyH2(TH2), 'r-')
%
% figure(2)
% plot(TNH3, PolyNH3(TNH3), 'r-')
%
% figure(3)
% plot(TN2, PolyN2(TN2), 'r-')
%
% figure(4)
% plot(TCH4, PolyCH4(TCH4), 'r-')
%
% figure(5)
% plot(THCN, PolyHCN(THCN), 'r-')
%
% figure(6)
% plot(TO2, PolyO2(TO2), 'r-')
%
% figure(7)
% plot(TCO2, PolyCO2(TCO2), 'r-')

end

%%
function nu = InterpolationKinematicViscosity(T, P)
% This function calculates the temperature dependenc of the viscosity by
% using the southerland equatoin

    R = 0.062363577;                                        %(m^3*torr)/(mol K)
    MW = [0.017, 0.016, 0.027, 0.02, 0.028, 0.032, 0.044];      %kg / mol
    MWrxn = sum(MW(1:5)) / 5;                               %average MW for the rxn
    MWcomb = sum(MW(5:7))/2;
%     mu0 = [3.19, 2.56, 21.1, 1.84, 3.78, 4.47, 3.74] .* 1e-5;   %in Pa*s
%     T0 = [600, 600, 20, 600, 600, 600, 600] + 273.15;           %in K
    rhorxn = (P * MWrxn) / (R * T);
    rhocomb =(P * MWcomb) / (R * T);

%     C = [370, 169, -, 72, 111, 127, 240];                       %Southerland constants in K
%     mu = mu0 .* ((T0 + C)./ (T + C)) .* (T / T0) .^ (3 / 2);    %Southerland formel

    mu = InterpolationDynamicViscositiy(T);

    nurxn = mu(1:5) ./ rhorxn;
    nucomb = mu(6:7) ./ rhocomb;
    nu = [nurxn; nucomb];
```

```
426 end
427
428 %%
429 function muint = InterpolationDynamicViscositiy(T)
430 %=========================================================================
431 % Data mu
432 %=========================================================================
433 TH2 = [0, 20, 50, 100, 200, 300, 400, 500, 600] + 273.15;               %temperature in K
434 nuH2 = [0.84, 0.88, 0.94, 1.04, 1.21, 1.37, 1.53, 1.69, 1.84] .* 1e-5;  %in W/(m*K)
435
436 TNH3 = [0, 20, 50, 100, 200, 300, 400, 500, 600] + 273.15;              %K
437 nuNH3 = [0.92, 0.99, 1.1, 1.3, 1.68, 2.06, 2.44, 2.82, 3.19] .* 1e-5;   %Pa*s
438
439 TN2 = [0, 20, 50, 100, 200, 300, 400, 500, 600] + 273.15;
440 nuN2 = [1.66, 1.76, 1.89, 2.12, 2.51, 2.86, 3.19, 3.49, 3.78] .* 1e-5;
441
442 TCH4 = [0, 20, 50, 100, 200, 300, 400, 500, 600] + 273.15;
443 nuCH4 = [1.03, 1.1, 1.19, 1.35, 1.63, 1.88, 2.11, 2.33, 2.53] .* 1e-5;
444
445 %NOT CORRECT!
446 THCN = [0, 20, 50, 100, 200, 300, 400, 500, 600] + 273.15;     %temperature in K
447 nuHCN = [1.37, 1.47, 1.61, 1.84, 2.29, 2.7, 3.07, 3.4, 3.7] .* 1e-5;    %(cal/(s*cm*K)) * 10^6
448
449 TO2 = [0, 20, 50, 100, 200, 300, 400, 500, 600] + 273.15;
450 nuO2 = [1.95, 2.04, 2.18, 2.44, 2.93, 3.37, 3.76, 4.13, 4.47] .* 1e-5;
451
452 TCO2 = [0, 20, 50, 100, 200, 300, 400, 500, 600] + 273.15;
453 nuCO2 = [1.37, 1.47, 1.61, 1.85, 2.3, 2.71, 3.08, 3.42, 3.74] .* 1e-5;
454
455 %=========================================================================
456 % Plots
457 %=========================================================================
458 % figure(1)
459 % hold on
460 % plot(TH2, nuH2, 'bx')
461 %
462 % figure(2)
463 % hold on
464 % plot(TNH3, nuNH3, 'bx')
465 %
466 % figure(3)
467 % hold on
468 % plot(TN2, nuN2, 'bx')
469 %
470 % figure(4)
471 % hold on
472 % plot(TCH4, nuCH4, 'bx')
473 %
474 % figure(5)
475 % hold on
476 % plot(THCN, nuHCN, 'bx')
477 %
478 % figure(6)
479 % hold on
480 % plot(TO2, nuO2, 'bx')
481 %
482 % figure(7)
483 % hold on
484 % plot(TCO2, nuCO2, 'bx')
485
486 %=========================================================================
487 % Linear fit
488 %=========================================================================
489 mdlH2 = polyfit(TH2, nuH2, 2);
490 bH2 = mdlH2(1);
491 mH2 = mdlH2(2);
492 uH2 = mdlH2(3);
493 PolyH2 = @(T) bH2 .* (T .^2) + mH2 .* T + uH2;
494 nuintH2 = PolyH2(T);
495
496 mdlNH3 = polyfit(TNH3, nuNH3, 2);
497 bNH3 = mdlNH3(1);
```

```
mNH3 = mdlNH3(2);
uNH3 = mdlNH3(3);
PolyNH3 = @(T) bNH3 .* (T .^2) + mNH3 .* T + uNH3;
nuintNH3 = PolyNH3(T);

mdlCH4 = polyfit(TCH4, nuCH4, 2);
bCH4 = mdlCH4(1);
mCH4 = mdlCH4(2);
uCH4 = mdlCH4(3);
PolyCH4 = @(T) bCH4 .* (T .^2) + mCH4 .* T + uCH4;
nuintCH4 = PolyCH4(T);

mdlHCN = polyfit(THCN, nuHCN, 2);
bHCN = mdlHCN(1);
mHCN = mdlHCN(2);
uHCN = mdlHCN(3);
PolyHCN = @(T) bHCN .* (T .^2) + mHCN .* T + uHCN;
nuintHCN = PolyHCN(T);

mdlN2 = polyfit(TN2, nuN2, 2);
bN2 = mdlN2(1);
mN2 = mdlN2(2);
uN2 = mdlN2(3);
PolyN2 = @(T) bN2 .* (T .^2) + mN2 .* T + uN2;
nuintN2 = PolyN2(T);

mdlO2 = polyfit(TO2, nuO2, 2);
bO2 = mdlO2(1);
mO2 = mdlO2(2);
uO2 = mdlO2(3);
PolyO2 = @(T) bO2 .* (T .^2) + mO2 .* T + uO2;
nuintO2 = PolyO2(T);

mdlCO2 = polyfit(TCO2, nuCO2, 2);
bCO2 = mdlCO2(1);
mCO2 = mdlCO2(2);
uCO2 = mdlCO2(3);
PolyCO2 = @(T) bCO2 .* (T .^2) + mCO2 .* T + uCO2;
nuintCO2 = PolyCO2(T);

muint = [nuintNH3, nuintCH4, nuintHCN, nuintH2, nuintN2, nuintO2, nuintCO2]';

%=========================================================================
% Plot Linear fit
%=========================================================================
% figure(1)
% plot(TH2, PolyH2(TH2), 'r-')
%
% figure(2)
% plot(TNH3, PolyNH3(TNH3), 'r-')
%
% figure(3)
% plot(TN2, PolyN2(TN2), 'r-')
%
% figure(4)
% plot(TCH4, PolyCH4(TCH4), 'r-')
%
% figure(5)
% plot(THCN, PolyHCN(THCN), 'r-')
%
% figure(6)
% plot(TO2, PolyO2(TO2), 'r-')
%
% figure(7)
% plot(TCO2, PolyCO2(TCO2), 'r-')

end

%%
function [Ap, Bp, Cp, Dp, Ep, Tcr] = ParameterHeatcapacity()
% CH4
A_CH4 = [-0.703029,  85.81217];
B_CH4 = [108.4773,  11.26467];
```

```
C_CH4 = [-42.52157,  -2.114146];
D_CH4 = [5.862788,  0.138190];
E_CH4 = [0.678565,  -26.42221];
T_CH4 = 1300; % [K] temperature, where the second parameter starts

% NH3
A_NH3 = [19.99563  ,52.02427];
B_NH3 = [49.77119,  18.48801];
C_NH3 = [-15.37599,  -3.765128];
D_NH3 = [1.921168  ,0.248541];
E_NH3 = [0.189174,  -12.45799];
T_NH3 = 1400;

% HCN
A_HCN = [32.69373  ,52.36527];
B_HCN = [22.59205,  5.563298];
C_HCN = [-4.369142,  -0.953224];
D_HCN = [-0.407697,  0.056711];
E_HCN = [-0.282399,  -7.564086];
T_HCN = 1300;

% H2
A_H2 = [33.066178,  18.563083];
B_H2 = [-11.363417,  12.257357];
C_H2 = [11.432816  ,-2.859786];
D_H2 = [-2.772874,  0.268238];
E_H2 = [-0.158558  ,1.977990];
T_H2 = 1000;

% N2 - still wrong!!
A_N2 = [19.50583,  19.50583];
B_N2 = [19.88705,  19.88705];
C_N2 = [-8.598535, -8.598535];
D_N2 = [1.369784,  1.369784];
E_N2 = [0.527601  , 0.527601];
T_N2 = 1000;

% Summariezed parameter for ODE
Ap = [A_CH4;A_NH3;A_HCN;A_H2;A_N2];
Bp = [B_CH4;B_NH3;B_HCN;B_H2;B_N2];
Cp = [C_CH4;C_NH3;C_HCN;C_H2;C_N2];
Dp = [D_CH4;D_NH3;D_HCN;D_H2;D_N2];
Ep = [E_CH4;E_NH3;E_HCN;E_H2;E_N2];
Tcr = [T_CH4,T_NH3,T_HCN,T_H2,T_N2];
end

%%
function molarvolume = Molarvolume(T, P, F, Ftot, R)

%     molefrac = F(1:5) ./ Ftot;                                    %molefraction (dimensionless)
%     [~, ~, ~, a, b] = PR_Parameterfunc(T, molefrac, P);     %parameter from PR equation
%     guess = (R * T) / P;                                    %guessed initial value for the molar volume at a
     temperature T
%
%     options = optimset('display', 'off');
%     vNH3 = lsqnonlin(@(v) (((R * T) / (v - b(1))) - (a(1) / (v^2 + 2 * v * b(1) - b(1)^2))) - P * molefrac(1)
     , guess, [], [], options);
%     vCH4 = lsqnonlin(@(v) (((R * T) / (v - b(2))) - (a(2) / (v^2 + 2 * v * b(2) - b(2)^2))) - P * molefrac(2)
     , guess, [], [], options);
%     vHCN = lsqnonlin(@(v) (((R * T) / (v - b(3))) - (a(3) / (v^2 + 2 * v * b(3) - b(3)^2))) - P * molefrac(3)
     , guess, [], [], options);
%     vH2 = lsqnonlin(@(v) (((R * T) / (v - b(4))) - (a(4) / (v^2 + 2 * v * b(4) - b(4)^2))) - P * molefrac(4),
      guess, [], [],options);
%     vN2 = lsqnonlin(@(v) (((R * T) / (v - b(5))) - (a(5) / (v^2 + 2 * v * b(5) - b(5)^2))) - P * molefrac(5),
      guess, [], [],options);
%
%     molarvolume = [vNH3; vCH4; vHCN; vH2; vN2];

 molefrac = F(1:5) ./ Ftot;                                    %molefraction (dimensionless)
    [~, ~, ~, a, b] = PR_Parameterfunc(T, molefrac, P);     %parameter from PR equation
    guess = (R * T) / P;                                    %guessed initial value for the molar volume at a
         temperature T
```

```
    options = optimset('display', 'off');
    Vm = lsqnonlin(@(v) (((R * T) / (v - b)) - (a / (v^2 + 2 * v * b - b^2))) - P, guess, [], [], options);

    vNH3 = Vm * molefrac(1);
    vCH4 = Vm * molefrac(2);
    vHCN = Vm * molefrac(3);
    vH2 = Vm * molefrac(4);
    vN2 = Vm * molefrac(5);

    molarvolume = [vNH3; vCH4; vHCN; vH2; vN2];
end
```

Listing 2: File used in section 3.1 to model the pressure dependence.

```
%%
% Case Study II
% Reactor Design
% Authors: Ramona Achermann, Tim Forster
% Zurich, 28.4.16

function CaseStudy_ReactorDesignMolefractionDependence
clear
close all
clc

%=========================================================================
% Define Constants
%=========================================================================
n = 400;                            %amount of tubes
L = 2;                              %length of pipe in m
l = 2.5e-3;                         %Wall thickness in m
D = 15e-3;                          %Inner diameter of pipe in m
A_cross = (D / 2)^2 * pi;           %Cross sectional Area in m^2
a = 4 / D;                          %Surface/Volume ratio of a cylindrical pipe
lambda = 4.5;                       %thermal conductivity W / (m*s)
alpha = lambda / l;                 %thermal transition coefficient in W / (m^2 * s)
H = [251e3, 91e3];                  %reaction enthalpies for HCN production (1) and side reaction (2) in J/mol
Ta = 1600;                          %Heating temperature in K
P = 760;                            %initial pressure in torr
T0 = 700;                           %initial temperature in K
NA = 6.02e23;                       %Avogadro number mol^-1
ratio = [1, 0.9, 0.85, 0.8, 0.75, 0.7];  %Different gas composition for the inlet

%=========================================================================
% Parameter for heat capacity
%=========================================================================
[Ap, Bp, Cp, Dp, Ep, Tcr] = ParameterHeatcapacity();

%=========================================================================
% Solve ODE
%=========================================================================
yield = zeros();
for i = 1 : 6

    FNH3 = 0.009;                           %mol/s
    FCH4 = ratio(i) * FNH3;                 %mol/s Ammonia 5% in excess
    initial = [FNH3, FCH4, 0, 0, 0, T0];    %initial fluxes in mol/s
    Vspan = [0, L * A_cross];               %integration range
    [V, x] = ode15s(@(V, x) func(V, x, a, alpha, Ta, H, NA, Ap, Bp, Cp, Dp, Ep, Tcr, P, A_cross), Vspan,
        initial);
    F = x(:, 1:5);                          %fluxes in mol/s
    z = V ./ A_cross;                       %length of pipe reactor
    yield(i) = F(end, 3) / F(1, 2);

    %Plot the concentration profile
    figure(1)
    hold on
    plot(z, F(:, 3) .* n)

    figure(2)
    hold on
    plot(z, F(:, 5) .* n)
```

```
end

    figure(1)
    legend(sprintf('HCN ratio_1=%g', ratio(1)), sprintf('HCN ratio_2=%g', ratio(2)), sprintf('HCN ratio_3=%g',
        ratio(3)), sprintf('HCN ratio_4=%g', ratio(4)), sprintf('HCN ratio_5=%g', ratio(5)), sprintf('HCN
        ratio_6=%g', ratio(6)), 'location', 'best')
    xlabel('z / m'), ylabel('F_{HCN} / (mol/s)')
    ax = gca;
    ax.YAxisLocation = 'left';
    ax.Box = 'on';
    saveas(figure(1),'MolefractionDependenceHCN','epsc');

    figure(2)
    legend(sprintf('N_2 ratio_1=%g', ratio(1)), sprintf('N_2 ratio_2=%g', ratio(2)), sprintf('N_2 ratio_3=%g',
        ratio(3)), sprintf('N_2 ratio_4=%g', ratio(4)), sprintf('N_2 ratio_5=%g', ratio(5)), sprintf('N_2
        ratio_6=%g', ratio(6)), 'location', 'best')
    xlabel('z / m'), ylabel('F_{N_2} / (mol/s)')
    ax = gca;
    ax.YAxisLocation = 'left';
    ax.Box = 'on';
    saveas(figure(2),'MolefractionDependenceN2','epsc');

%more ratios
ratio = [1.5, 1.3, 1.1, 1.05, 1, 0.95, 0.9, 0.85, 0.8, 0.75, 0.7]; %ratios of CH4/NH3
for i = 1 : length(ratio)

    FNH3 = 0.009;                           %mol/s
    FCH4 = ratio(i) * FNH3;                     %mol/s Ammonia 5% in excess
    initial = [FNH3, FCH4, 0, 0, 0, T0];    %initial fluxes in mol/s
    Vspan = [0, L * A_cross];               %integration range
    [~, x] = ode15s(@(V, x) func(V, x, a, alpha, Ta, H, NA, Ap, Bp, Cp, Dp, Ep, Tcr, P, A_cross), Vspan,
        initial);
    F = x(:, 1:5);                          %fluxes in mol/s
    yield(i) = F(end, 3) / F(1, 2);

end

figure(3)
hold on
plot(ratio, yield, 'r.-')
xlabel('Ratio CH_4 / NH_3'), ylabel('Yield / [-]')
ax = gca;
ax.YAxisLocation = 'left';
ax.Box = 'on';
saveas(figure(3),'MolfractionDependenceYield','epsc');

end

%%
function dx = func(~, x, a, alpha, Ta, H, NA, Ap, Bp, Cp, Dp, Ep, Tcr, P0, A_cross)
% Function that defines the ODEs and calculates the heat capacities at
% every temperature in the reactor
% The variables are:
    % x(1)          ...NH3
    % x(2)          ...CH4
    % x(3)          ...HCN
    % x(4)          ...H2
    % x(5)          ...N2
    % x(6)          ...T

% Molar heat capacities of each species at each point/temperature in reactor
    cP = zeros();
    for i = 1:5 % Do it for all 5 species
        t = x(6) / 1000;
        if x(6) < Tcr(i)
            j = 1;
        else
            j = 2;
        end
        cP(i) = Ap(i,j) + Bp(i,j)*t + Cp(i,j)*t^2 + Dp(i,j)*t^3 + Ep(i,j)/t^2;
    end
```

```
    cP_CH4 = cP(1);
    cP_NH3 = cP(2);
    cP_HCN = cP(3);
    cP_H2 = cP(4);
    cP_N2 = cP(5);

%Total flux is defined as:
    Ftot = x(1) + x(2) + x(3) + x(4) + x(5);

%Calculate parameter for PR equation
    T = x(6);
    molefrac = x ./ Ftot;
    [Amix, Bmix, ~, ~, ~] = PR_Parameterfunc(T, molefrac, P0);

%Solve PR EoS
    phi = PR_EoS(Amix, Bmix);
    P = phi * P0;

%Calculate lambda for the certain temperature
    lambda = InterpolationLambda(x(6));
    lambdamix_rxn = sum((x(1:5) ./ Ftot) .* lambda(1:5));

%Calculate viscosity (mu) for the certain temperature
    mu = InterpolationDynamicViscositiy(T);
    mumix_rxn = sum((x(1:5) ./ Ftot) .* mu(1:5));

%Calculate viscosity (nu) for the certain temperature
    nu = InterpolationKinematicViscosity(T, P);
    numix_rxn = sum((x(1:5) ./ Ftot) .* nu(1:5));

%Calculate the gas velocity in the pipe (ideal)
    R = 0.062363577;                            %(m^3*torr)/(mol K)
%     u_ideal = (Ftot * R * x(6)) / (P * A_cross);

%Calculate the gas velocity in the pipe over correction of the molar
%volumes by PR equation
    molarvolume = Molarvolume(T, P, x, Ftot, R);
    Q_real = sum(molarvolume) * Ftot;
    u_real = Q_real / A_cross;

%Calculate Reynolds number for the gas in the tube (rxn-gas)
    D = 15e-3;
    Re_rxn = (u_real * D) / numix_rxn;

%Calculate Prandtl number for the gas in the tube (rxn-gas)
%     cP_mix = sum((x(1) / Ftot) * cP_NH3 + (x(2) / Ftot) * cP_CH4 + ...
%             (x(3) / Ftot) * cP_HCN + (x(4) / Ftot) * cP_H2 + (x(5) / Ftot) * cP_N2);
%     Pr_rxn= (cP_mix * mumix_rxn) / lambdamix_rxn;
     Pr_rxn = 0.79;  %ASSUMPTION FROM BOOK

%Calculate Nusselt number for the gas in the tube (rxn-gas) with two
%different approaches
%     Nu_rxn = 0.02 .* (Re_rxn .^ 0.8) .* (Pr_rxn .^ 0.43);
    mu_end = InterpolationDynamicViscositiy(T);
    mumix_rxn_end = sum((x(1:5) ./ Ftot) .* mu_end(1:5));
    Nu_rxn = 0.027 * (Re_rxn^0.8) * (Pr_rxn^0.333) * ((mumix_rxn / mumix_rxn_end)^0.14);
    alpha_rxn = (Nu_rxn .* lambdamix_rxn) / D;

%Calculate overall heat transfer coefficient
    U = 1 / ((1 / alpha) + (1 / alpha_rxn));

%The rate expressions are defined separately in functions:
    rHCN = rateHCN(x(1), x(2), x(6), Ftot, NA, P);
    rN2 = rateN2(x(1), x(2), x(6), Ftot, NA, P);

%The mass balances and energy balance are defined as:
    dx(1) = - rHCN * a - 2 * rN2 * a;
    dx(2) = - rHCN * a;
    dx(3) = rHCN * a;
    dx(4) = 3 * rHCN * a + 3 * rN2 * a;
    dx(5) = rN2 * a;
    dx(6) = (U * a * (Ta - x(6)) - (rHCN * a *  H(1) + rN2 * a * H(2))) / (x(1) * cP_NH3 + x(2) * cP_CH4 + x(3)
          * cP_HCN + x(4) * cP_H2 + x(5) * cP_N2);
```

```
%Sum these up and send them back to the main file:
    dx = [dx(1); dx(2); dx(3); dx(4); dx(5); dx(6)];

end

%%
function rHCN = rateHCN(x1, x2, x6, Ftot, NA, P)
% rate expression of the reaction that produces HCN
% ATTENTION: units are mol / (s * m^2)

    rHCN =  (1e4 / NA) * ((7.8e18 * exp(-1950 / x6) * ((x2 * P) / Ftot) * (((x1 * P) / Ftot) ^ 0.5)) / ...
            ((1 + 0.044 * exp(2390 / x6) * ((x2 * P) / Ftot) * (((x1 * P) / Ftot) ^-0.5))^4));

end

%%
function rN2 = rateN2(x1, x2, x6, Ftot, NA, P)
% rate expression of the reaction that produces N2
% ATTENTION: units are mol / (s * m^2)

    rN2 =   (1e4 / NA) * ((4.9e18 * exp(-2130 / x6) * ((x1 * P) / Ftot)) / ...
            ((1 + 0.044 * exp(2390 / x6) * ((x2 * P) / Ftot) * (((x1 * P) / Ftot) ^ -0.5))^3));

end

%%
function [Amix, Bmix, E, amix, bmix] = PR_Parameterfunc(T, x, P0)
% This function calculates the parameter for the peng robinson equation
% 1: NH3
% 2: CH4
% 3: HCN
% 4: H2
% 5: N2

% T: temperature in K
% x: mole fractions (dimensionless)

    R = 0.062363577;                                        %(m^3*torr)/(mol K)
    P = P0 ;                                                %ideal pressure in torr
    Tc = [132.25, -82.59, 183.5, -240.01, -146.96] + 273.15;    %critical temperatures in K
    pc = [113.3, 45.99, 53.9, 12.96, 33.96] .* 750.06;              %critical pressures in torr

    Tr = T ./ Tc;                                           %reduced temperatures (dimensionless)
    s = [0.37464, 1.54226, -0.26992];
    omega = [0.25 , 0.011, 0.4095, -0.216, 0.039];              %acentric factors (dimensionless)
    omegamix =  x(1) .* omega(1) + x(2) .* omega(2) + x(3) .* omega(3) + ...
                x(4) .* omega(4) + x(5) .* omega(5);

    S = s(1) + s(2) .* omegamix + s(3) .* omegamix .^ 2;
    k = (1 + S .* (1 - sqrt(Tr))) .^ 2;
    E = S .* sqrt(Tr ./ k);

    a = (0.457235 .* (R ^ 2) .* (Tc .^ 2) .* k) ./ pc;
    b = (0.07780 .* R .* Tc) ./ (pc);

    amix = (x(1) .* sqrt(a(1)) + x(2) .* sqrt(a(2)) + ...
            x(3) .* sqrt(a(3)) + x(4) .* sqrt(a(4)) + ...
            x(5) .* sqrt(a(5))) .^ 2;

    bmix =  x(1) .* b(1) + x(2) .* b(2) + x(3) .* b(3) + ...
            x(4) .* b(4) + x(5) .* b(5);

    Amix = (amix .* P) ./ ((R .* T) .^ 2);
    Bmix = (bmix .* P) ./ (R .* T);

end

%%
function phi = PR_EoS(Amix, Bmix)

    coef = [1, (-1 + Bmix), (Amix - 2 * Bmix - 3 * Bmix^2), (-Amix * Bmix + Bmix^2 + Bmix^3)];
    Z = roots(coef);
```

```
271     Z = max(Z);
272     phi = exp(Z - 1 - (Amix  / (2 * sqrt(2) * Bmix)) * log( ((Z + Bmix * (1 + sqrt(2)))) / (Z + Bmix * (1 -
            sqrt(2)))) - log(Z - Bmix));

274 end

276 %%
277 function lambdaint = InterpolationLambda(T)
278 % This function calculates the temperature dependence of the thermal
279 % conductivity by a linear fit and extrapolation (this is a plausible
280 % method --> see source in report for validity)

282 %=========================================================================
283 % Data thermal conductivity
284 %=========================================================================
285 TH2 = [100, 200, 300, 400];                         %temperature in K
286 lambdaH2 = [68.6, 131.7, 186.9, 230.4] ./ 1000;     %in W/(m*K)

288 TNH3 = [300, 400, 500, 600];
289 lambdaNH3 = [24.4, 37.4, 51.6, 66.8] ./ 1000;

291 TN2 = [100, 200, 300, 400, 500, 600];
292 lambdaN2 = [9.8, 18.7, 26.0, 32.3, 38.3, 44.0] ./ 1000;

294 TCH4 = [200, 300, 400, 500, 600];
295 lambdaCH4 = [22.5, 34.1, 49.1, 66.5, 84.1] ./ 1000;

297 THCN = [-17.8, -6.7, 4.4, 15.6, 26.7, 37.8, 48.9] + 273.15;     %temperature in K
298 lambdaHCN = [23.97, 25.62, 26.86, 28.1, 29.75, 30.99, 32.6] ./ 1000;    %(cal/(s*cm*K)) * 10^6

300 TO2 = [100, 200, 300, 400, 500, 600];
301 lambdaO2 = [9.3, 18.4, 26.3, 33.7, 41.0, 48.1] ./ 1000;

303 TCO2 = [200, 300, 400, 500, 600];
304 lambdaCO2 = [9.6, 16.8, 25.1, 33.5, 41.6] ./ 1000;

306 %=========================================================================
307 % Plots
308 %=========================================================================
309 % figure(1)
310 % hold on
311 % plot(TH2, lambdaH2, 'bx')
312 %
313 % figure(2)
314 % hold on
315 % plot(TNH3, lambdaNH3, 'bx')
316 %
317 % figure(3)
318 % hold on
319 % plot(TN2, lambdaN2, 'bx')
320 %
321 % figure(4)
322 % hold on
323 % plot(TCH4, lambdaCH4, 'bx')
324 %
325 % figure(5)
326 % hold on
327 % plot(THCN, lambdaHCN, 'bx')
328 %
329 % figure(6)
330 % hold on
331 % plot(TO2, lambdaO2, 'bx')
332 %
333 % figure(7)
334 % hold on
335 % plot(TCO2, lambdaCO2, 'bx')

337 %=========================================================================
338 % Linear fit
339 %=========================================================================
340 mdlH2 = fitlm(TH2, lambdaH2);
341 bH2 = mdlH2.Coefficients{1,1};
342 mH2 = mdlH2.Coefficients{2,1};
```

```
PolyH2 = @(T) mH2 * T + bH2;
lambdaintH2 = PolyH2(T);

mdlNH3 = fitlm(TNH3, lambdaNH3);
bNH3 = mdlNH3.Coefficients{1,1};
mNH3 = mdlNH3.Coefficients{2,1};
PolyNH3 = @(T) mNH3 * T + bNH3;
lambdaintNH3 = PolyNH3(T);

mdlN2 = fitlm(TN2, lambdaN2);
bN2 = mdlN2.Coefficients{1,1};
mN2 = mdlN2.Coefficients{2,1};
PolyN2 = @(T) mN2 * T + bN2;
lambdaintN2 = PolyN2(T);

mdlCH4 = fitlm(TCH4, lambdaCH4);
bCH4 = mdlCH4.Coefficients{1,1};
mCH4 = mdlCH4.Coefficients{2,1};
PolyCH4 = @(T) mCH4 * T + bCH4;
lambdaintCH4 = PolyCH4(T);

mdlHCN = fitlm(THCN, lambdaHCN);
bHCN = mdlHCN.Coefficients{1,1};
mHCN = mdlHCN.Coefficients{2,1};
PolyHCN = @(T) mHCN * T + bHCN;
lambdaintHCN = PolyHCN(T);

mdlO2 = fitlm(TO2, lambdaO2);
bO2 = mdlO2.Coefficients{1,1};
mO2 = mdlO2.Coefficients{2,1};
PolyO2 = @(T) mO2 * T + bO2;
lambdaintO2 = PolyO2(T);

mdlCO2 = fitlm(TCO2, lambdaCO2);
bCO2 = mdlCO2.Coefficients{1,1};
mCO2 = mdlCO2.Coefficients{2,1};
PolyCO2 = @(T) mCO2 * T + bCO2;
lambdaintCO2 = PolyCO2(T);

lambdaint = [lambdaintNH3, lambdaintCH4, lambdaintHCN, lambdaintH2, lambdaintN2, lambdaintO2, lambdaintCO2]';

%=========================================================================
% Plot Linear fit
%=========================================================================
% figure(1)
% plot(TH2, PolyH2(TH2), 'r-')
%
% figure(2)
% plot(TNH3, PolyNH3(TNH3), 'r-')
%
% figure(3)
% plot(TN2, PolyN2(TN2), 'r-')
%
% figure(4)
% plot(TCH4, PolyCH4(TCH4), 'r-')
%
% figure(5)
% plot(THCN, PolyHCN(THCN), 'r-')
%
% figure(6)
% plot(TO2, PolyO2(TO2), 'r-')
%
% figure(7)
% plot(TCO2, PolyCO2(TCO2), 'r-')

end

%%
function nu = InterpolationKinematicViscosity(T, P)
% This function calculates the temperature dependenc of the viscosity by
% using the southerland equatoin

    R = 0.062363577;                                        %(m^3*torr)/(mol K)
```

```
    MW = [0.017, 0.016, 0.027, 0.02, 0.028, 0.032, 0.044];      %kg / mol
    MWrxn = sum(MW(1:5)) / 5;                                   %average MW for the rxn
    MWcomb = sum(MW(5:7))/2;
%     mu0 = [3.19, 2.56, 21.1, 1.84, 3.78, 4.47, 3.74] .* 1e-5;   %in Pa*s
%     T0 = [600, 600, 20, 600, 600, 600, 600] + 273.15;           %in K
    rhorxn = (P * MWrxn) / (R * T);
    rhocomb =(P * MWcomb) / (R * T);

%     C = [370, 169, -, 72, 111, 127, 240];                       %Southerland constants in K
%     mu = mu0 .* ((T0 + C)./ (T + C)) .* (T / T0) .^ (3 / 2);    %Southerland formel

    mu = InterpolationDynamicViscositiy(T);

    nurxn = mu(1:5) ./ rhorxn;
    nucomb = mu(6:7) ./ rhocomb;
    nu = [nurxn; nucomb];

end

%%
function muint = InterpolationDynamicViscositiy(T)
%==========================================================================
% Data mu
%==========================================================================
TH2 = [0, 20, 50, 100, 200, 300, 400, 500, 600] + 273.15;               %temperature in K
nuH2 = [0.84, 0.88, 0.94, 1.04, 1.21, 1.37, 1.53, 1.69, 1.84] .* 1e-5;  %in W/(m*K)

TNH3 = [0, 20, 50, 100, 200, 300, 400, 500, 600] + 273.15;              %K
nuNH3 = [0.92, 0.99, 1.1, 1.3, 1.68, 2.06, 2.44, 2.82, 3.19] .* 1e-5;   %Pa*s

TN2 = [0, 20, 50, 100, 200, 300, 400, 500, 600] + 273.15;
nuN2 = [1.66, 1.76, 1.89, 2.12, 2.51, 2.86, 3.19, 3.49, 3.78] .* 1e-5;

TCH4 = [0, 20, 50, 100, 200, 300, 400, 500, 600] + 273.15;
nuCH4 = [1.03, 1.1, 1.19, 1.35, 1.63, 1.88, 2.11, 2.33, 2.53] .* 1e-5;

%NOT CORRECT!
THCN = [0, 20, 50, 100, 200, 300, 400, 500, 600] + 273.15;     %temperature in K
nuHCN = [1.37, 1.47, 1.61, 1.84, 2.29, 2.7, 3.07, 3.4, 3.7] .* 1e-5;    %(cal/(s*cm*K)) * 10^6

TO2 = [0, 20, 50, 100, 200, 300, 400, 500, 600] + 273.15;
nuO2 = [1.95, 2.04, 2.18, 2.44, 2.93, 3.37, 3.76, 4.13, 4.47] .* 1e-5;

TCO2 = [0, 20, 50, 100, 200, 300, 400, 500, 600] + 273.15;
nuCO2 = [1.37, 1.47, 1.61, 1.85, 2.3, 2.71, 3.08, 3.42, 3.74] .* 1e-5;

%==========================================================================
% Plots
%==========================================================================
% figure(1)
% hold on
% plot(TH2, nuH2, 'bx')
%
% figure(2)
% hold on
% plot(TNH3, nuNH3, 'bx')
%
% figure(3)
% hold on
% plot(TN2, nuN2, 'bx')
%
% figure(4)
% hold on
% plot(TCH4, nuCH4, 'bx')
%
% figure(5)
% hold on
% plot(THCN, nuHCN, 'bx')
%
% figure(6)
% hold on
% plot(TO2, nuO2, 'bx')
%
```

```
% figure(7)
% hold on
% plot(TCO2, nuCO2, 'bx')

%==========================================================================
% Linear fit
%==========================================================================
mdlH2 = polyfit(TH2, nuH2, 2);
bH2 = mdlH2(1);
mH2 = mdlH2(2);
uH2 = mdlH2(3);
PolyH2 = @(T) bH2 .* (T .^2) + mH2 .* T + uH2;
nuintH2 = PolyH2(T);

mdlNH3 = polyfit(TNH3, nuNH3, 2);
bNH3 = mdlNH3(1);
mNH3 = mdlNH3(2);
uNH3 = mdlNH3(3);
PolyNH3 = @(T) bNH3 .* (T .^2) + mNH3 .* T + uNH3;
nuintNH3 = PolyNH3(T);

mdlCH4 = polyfit(TCH4, nuCH4, 2);
bCH4 = mdlCH4(1);
mCH4 = mdlCH4(2);
uCH4 = mdlCH4(3);
PolyCH4 = @(T) bCH4 .* (T .^2) + mCH4 .* T + uCH4;
nuintCH4 = PolyCH4(T);

mdlHCN = polyfit(THCN, nuHCN, 2);
bHCN = mdlHCN(1);
mHCN = mdlHCN(2);
uHCN = mdlHCN(3);
PolyHCN = @(T) bHCN .* (T .^2) + mHCN .* T + uHCN;
nuintHCN = PolyHCN(T);

mdlN2 = polyfit(TN2, nuN2, 2);
bN2 = mdlN2(1);
mN2 = mdlN2(2);
uN2 = mdlN2(3);
PolyN2 = @(T) bN2 .* (T .^2) + mN2 .* T + uN2;
nuintN2 = PolyN2(T);

mdlO2 = polyfit(TO2, nuO2, 2);
bO2 = mdlO2(1);
mO2 = mdlO2(2);
uO2 = mdlO2(3);
PolyO2 = @(T) bO2 .* (T .^2) + mO2 .* T + uO2;
nuintO2 = PolyO2(T);

mdlCO2 = polyfit(TCO2, nuCO2, 2);
bCO2 = mdlCO2(1);
mCO2 = mdlCO2(2);
uCO2 = mdlCO2(3);
PolyCO2 = @(T) bCO2 .* (T .^2) + mCO2 .* T + uCO2;
nuintCO2 = PolyCO2(T);

muint = [nuintNH3, nuintCH4, nuintHCN, nuintH2, nuintN2, nuintO2, nuintCO2]';

%==========================================================================
% Plot Linear fit
%==========================================================================
% figure(1)
% plot(TH2, PolyH2(TH2), 'r-')
%
% figure(2)
% plot(TNH3, PolyNH3(TNH3), 'r-')
%
% figure(3)
% plot(TN2, PolyN2(TN2), 'r-')
%
% figure(4)
% plot(TCH4, PolyCH4(TCH4), 'r-')
%
```

```
% figure(5)
% plot(THCN, PolyHCN(THCN), 'r-')
%
% figure(6)
% plot(TO2, PolyO2(TO2), 'r-')
%
% figure(7)
% plot(TCO2, PolyCO2(TCO2), 'r-')

end

%%
function [Ap, Bp, Cp, Dp, Ep, Tcr] = ParameterHeatcapacity()
% CH4
A_CH4 = [-0.703029,  85.81217];
B_CH4 = [108.4773,  11.26467];
C_CH4 = [-42.52157,  -2.114146];
D_CH4 = [5.862788,  0.138190];
E_CH4 = [0.678565,  -26.42221];
T_CH4 = 1300; % [K] temperature, where the second parameter starts

% NH3
A_NH3 = [19.99563  ,52.02427];
B_NH3 = [49.77119,  18.48801];
C_NH3 = [-15.37599,  -3.765128];
D_NH3 = [1.921168  ,0.248541];
E_NH3 = [0.189174,  -12.45799];
T_NH3 = 1400;

% HCN
A_HCN = [32.69373  ,52.36527];
B_HCN = [22.59205,  5.563298];
C_HCN = [-4.369142,  -0.953224];
D_HCN = [-0.407697,  0.056711];
E_HCN = [-0.282399,  -7.564086];
T_HCN = 1300;

% H2
A_H2 = [33.066178,  18.563083];
B_H2 = [-11.363417,  12.257357];
C_H2 = [11.432816  ,-2.859786];
D_H2 = [-2.772874,  0.268238];
E_H2 = [-0.158558  ,1.977990];
T_H2 = 1000;

% N2 - still wrong!!
A_N2 = [19.50583,  19.50583];
B_N2 = [19.88705,  19.88705];
C_N2 = [-8.598535, -8.598535];
D_N2 = [1.369784,  1.369784];
E_N2 = [0.527601  , 0.527601];
T_N2 = 1000;

% Summariezed parameter for ODE
Ap = [A_CH4;A_NH3;A_HCN;A_H2;A_N2];
Bp = [B_CH4;B_NH3;B_HCN;B_H2;B_N2];
Cp = [C_CH4;C_NH3;C_HCN;C_H2;C_N2];
Dp = [D_CH4;D_NH3;D_HCN;D_H2;D_N2];
Ep = [E_CH4;E_NH3;E_HCN;E_H2;E_N2];
Tcr = [T_CH4,T_NH3,T_HCN,T_H2,T_N2];
end

%%
function molarvolume = Molarvolume(T, P, F, Ftot, R)

%     molefrac = F(1:5) ./ Ftot;                                    %molefraction (dimensionless)
%     [~, ~, ~, a, b] = PR_Parameterfunc(T, molefrac, P);     %parameter from PR equation
%     guess = (R * T) / P;                                    %guessed initial value for the molar volume at a
    temperature T
%
%     options = optimset('display', 'off');
%     vNH3 = lsqnonlin(@(v) (((R * T) / (v - b(1))) - (a(1) / (v^2 + 2 * v * b(1) - b(1)^2))) - P * molefrac(1)
    , guess, [], [], options);
```

```
%     vCH4 = lsqnonlin(@(v) (((R * T) / (v - b(2))) - (a(2) / (v^2 + 2 * v * b(2) - b(2)^2))) - P * molefrac(2)
      , guess, [], [], options);
%     vHCN = lsqnonlin(@(v) (((R * T) / (v - b(3))) - (a(3) / (v^2 + 2 * v * b(3) - b(3)^2))) - P * molefrac(3)
      , guess, [], [], options);
%     vH2 = lsqnonlin(@(v) (((R * T) / (v - b(4))) - (a(4) / (v^2 + 2 * v * b(4) - b(4)^2))) - P * molefrac(4),
      guess, [], [],options);
%     vN2 = lsqnonlin(@(v) (((R * T) / (v - b(5))) - (a(5) / (v^2 + 2 * v * b(5) - b(5)^2))) - P * molefrac(5),
      guess, [], [],options);
%
%     molarvolume = [vNH3; vCH4; vHCN; vH2; vN2];

 molefrac = F(1:5) ./ Ftot;                                  %molefraction (dimensionless)
    [~, ~, ~, a, b] = PR_Parameterfunc(T, molefrac, P);     %parameter from PR equation
    guess = (R * T) / P;                                    %guessed initial value for the molar volume at a
         temperature T

    options = optimset('display', 'off');
    Vm = lsqnonlin(@(v) (((R * T) / (v - b)) - (a / (v^2 + 2 * v * b - b^2))) - P, guess, [], [], options);

    vNH3 = Vm * molefrac(1);
    vCH4 = Vm * molefrac(2);
    vHCN = Vm * molefrac(3);
    vH2 = Vm * molefrac(4);
    vN2 = Vm * molefrac(5);

    molarvolume = [vNH3; vCH4; vHCN; vH2; vN2];
end
```

Listing 3: File used in section 3.1 to model the dependence on the molefraction of the inlet feed.

```
%%
% Case Study II
% Reactor Design
% Authors: Ramona Achermann, Tim Forster
% Zurich, 25.4.16

function CaseStudy_ReactorDesignFeedDependence
clear
close all
clc

%=========================================================================
% Define Constants
%=========================================================================
n = 400;                            %amount of tubes
L = 2;                              %length of pipe in m
l = 2.5e-3;                         %Wall thickness in m
D = 15e-3;                          %Inner diameter of pipe in m
A_cross = (D / 2)^2 * pi;           %Cross sectional Area in m^2
a = 4 / D;                          %Surface/Volume ratio of a cylindrical pipe
lambda = 4.5;                       %thermal conductivity W / (m*s)
alpha = lambda / l;                 %thermal transition coefficient in W / (m^2 * s)
H = [251e3, 91e3];                  %reaction enthalpies for HCN production (1) and side reaction (2) in J/mol
Ta = 1600;                          %Heating temperature in K
P = 760;                            %initial pressure in torr
T0 = 700;                           %initial temperature in K
NA = 6.02e23;                       %Avogadro number mol^-1
FNH3 = [0.01, 0.015, 0.02, 0.03, 0.04, 0.05];  %Different gas composition for the inlet

%=========================================================================
% Parameter for heat capacity
%=========================================================================
[Ap, Bp, Cp, Dp, Ep, Tcr] = ParameterHeatcapacity();

%=========================================================================
% Solve ODE
%=========================================================================
yield = zeros();
for i = 1 : 6

    FCH4 = 0.95 * FNH3(i);                %mol/s Ammonia 5% in excess
    initial = [FNH3(i), FCH4, 0, 0, 0, T0];    %initial fluxes in mol/s
```

```
43      Vspan = [0, L * A_cross];               %integration range
44      [V, x] = ode15s(@(V, x) func(V, x, a, alpha, Ta, H, NA, Ap, Bp, Cp, Dp, Ep, Tcr, P), Vspan, initial);
45      F = x(:, 1:5);                          %fluxes in mol/s
46      z = V ./ A_cross;                       %length of pipe reactor
47      yield(i) = F(end, 3) / F(1, 2);

49      %Plot the concentration profile
50      figure(1)
51      hold on
52      plot(z, F(:, 3) .* n)


55  end

57      figure(1)
58      legend(sprintf('F_{NH_3}=%g', FNH3(1)), sprintf('F_{NH_3}=%g', FNH3(2)), sprintf('F_{NH_3}=%g', FNH3(3)),
            sprintf('F_{NH_3}=%g', FNH3(4)), sprintf('F_{NH_3}=%g', FNH3(5)), sprintf('F_{NH_3}=%g', FNH3(6)), '
            location', 'best')
59      xlabel('z / m'), ylabel('F_{HCN} / (mol/s)')
60      ax = gca;
61      ax.YAxisLocation = 'left';
62      ax.Box = 'on';
63      saveas(figure(1),'FeedDependenceHCN','epsc');

65      figure(2)
66      plot(FNH3 .* n, yield, 'rx-')
67      xlabel('Inlet flow rate of NH_3 / (mol/s)'), ylabel('Yield / [-]')
68      ax = gca;
69      ax.YAxisLocation = 'left';
70      ax.Box = 'on';
71      saveas(figure(2),'FeedDependenceYield','epsc')

73      figure(3)
74          left_color = [0 0 0];
75          right_color = [0 0 0];
76          set(figure(3),'defaultAxesColorOrder',[left_color; right_color]);
77      hold on
78      ax = gca;
79      ax.YAxisLocation = 'right';
80      ax.Box = 'on';
81      yyaxis 'right'

83  %Plot yield curve with fit
84      p = polyfit(FNH3*n, yield, 2);
85      func = @(f) p(1) .* f .^2 + p(2) .* f + p(3);
86      fvec = linspace(FNH3(1)*n, FNH3(end)*n, 50);
87      plot(fvec, func(fvec), 'r.-')
88      xlabel('Inlet flow rate of NH_3 / (mol/s)'), ylabel('Yield / [-]')
89      yyaxis 'left'

91  %Plot outlet curve with fit
92      p = polyfit(FNH3*n, outlet, 2);
93      func = @(f) p(1) .* f .^2 + p(2) .* f + p(3);
94      fvec = linspace(FNH3(1)*n, FNH3(end)*n, 50);
95      plot(fvec, func(fvec)*n, 'b.-')
96          line([12.8 12.8], ylim, 'Color', [0 0 0]);
97      ylabel('Outlet flow rate HCN / (mol/s) ')
98      legend('Outlet of HCN', 'Optimal flow rate', 'Yield', 'location', 'southwest')
99      saveas(figure(3),'FeedDependenceYieldAndOutlet','epsc');

101 end

103 %%
104 function [Ap, Bp, Cp, Dp, Ep, Tcr] = ParameterHeatcapacity()
105 % CH4
106 A_CH4 = [-0.703029,  85.81217];
107 B_CH4 = [108.4773,  11.26467];
108 C_CH4 = [-42.52157,  -2.114146];
109 D_CH4 = [5.862788,  0.138190];
110 E_CH4 = [0.678565,  -26.42221];
111 T_CH4 = 1300; % [K] temperature, where the second parameter starts

113 % NH3
```

```
A_NH3 = [19.99563  ,52.02427];
B_NH3 = [49.77119,  18.48801];
C_NH3 = [-15.37599,  -3.765128];
D_NH3 = [1.921168  ,0.248541];
E_NH3 = [0.189174,  -12.45799];
T_NH3 = 1400;

% HCN
A_HCN = [32.69373  ,52.36527];
B_HCN = [22.59205,  5.563298];
C_HCN = [-4.369142,  -0.953224];
D_HCN = [-0.407697,  0.056711];
E_HCN = [-0.282399,  -7.564086];
T_HCN = 1300;

% H2
A_H2 = [33.066178,  18.563083];
B_H2 = [-11.363417,  12.257357];
C_H2 = [11.432816  ,-2.859786];
D_H2 = [-2.772874,  0.268238];
E_H2 = [-0.158558  ,1.977990];
T_H2 = 1000;

% N2 - still wrong!!
A_N2 = [19.50583,  19.50583];
B_N2 = [19.88705,  19.88705];
C_N2 = [-8.598535, -8.598535];
D_N2 = [1.369784,  1.369784];
E_N2 = [0.527601  , 0.527601];
T_N2 = 1000;

% Summariezed parameter for ODE
Ap = [A_CH4;A_NH3;A_HCN;A_H2;A_N2];
Bp = [B_CH4;B_NH3;B_HCN;B_H2;B_N2];
Cp = [C_CH4;C_NH3;C_HCN;C_H2;C_N2];
Dp = [D_CH4;D_NH3;D_HCN;D_H2;D_N2];
Ep = [E_CH4;E_NH3;E_HCN;E_H2;E_N2];
Tcr = [T_CH4,T_NH3,T_HCN,T_H2,T_N2];
end

%%
function dx = func(~, x, a, alpha, Ta, H, NA, Ap, Bp, Cp, Dp, Ep, Tcr, P)
% Function that defines the ODEs and calculates the heat capacities at
% every temperature in the reactor
% The variables are:
    % x(1)          ...NH3
    % x(2)          ...CH4
    % x(3)          ...HCN
    % x(4)          ...H2
    % x(5)          ...N2
    % x(6)          ...T

% Molar heat capacities of each species at each point/temperature in reactor
    cP = zeros();
    for i = 1:5 % Do it for all 5 species
        t = x(6) / 1000;
        if x(6) < Tcr(i)
            j = 1;
        else
            j = 2;
        end
        cP(i) = Ap(i,j) + Bp(i,j)*t + Cp(i,j)*t^2 + Dp(i,j)*t^3 + Ep(i,j)/t^2;
    end

    cP_CH4 = cP(1);
    cP_NH3 = cP(2);
    cP_HCN = cP(3);
    cP_H2 = cP(4);
    cP_N2 = cP(5);

%Total flux is defined as:
    Ftot = x(1) + x(2) + x(3) + x(4) + x(5);
```

```
%The rate expressions are defined separately in functions:
    rHCN = rateHCN(x(1), x(2), x(6), Ftot, NA, P);
    rN2 = rateN2(x(1), x(2), x(6), Ftot, NA, P);

%The mass balances and energy balance are defined as:
    dx(1) = - rHCN * a - 2 * rN2 * a;
    dx(2) = - rHCN * a;
    dx(3) = rHCN * a;
    dx(4) = 3 * rHCN * a + 3 * rN2 * a;
    dx(5) = rN2 * a;
    dx(6) = (alpha * a * (Ta - x(6)) - (rHCN * a *  H(1) + rN2 * a * H(2))) / (x(1) * cP_NH3 + x(2) * cP_CH4 +
         x(3) * cP_HCN + x(4) * cP_H2 + x(5) * cP_N2);

%Sum these up and send them back to the main file:
    dx = [dx(1); dx(2); dx(3); dx(4); dx(5); dx(6)];

end

%%
function rHCN = rateHCN(x1, x2, x6, Ftot, NA, P)
% rate expression of the reaction that produces HCN
% ATTENTION: units are mol / (s * m^2)

    rHCN =  (1e4 / NA) * ((7.8e18 * exp(-1950 / x6) * ((x2 * P) / Ftot) * (((x1 * P) / Ftot) ^ 0.5)) / ...
            ((1 + 0.044 * exp(2390 / x6) * ((x2 * P) / Ftot) * (((x1 * P) / Ftot) ^-0.5))^4));

end

%%
function rN2 = rateN2(x1, x2, x6, Ftot, NA, P)
% rate expression of the reaction that produces N2
% ATTENTION: units are mol / (s * m^2)

    rN2 =   (1e4 / NA) * ((4.9e18 * exp(-2130 / x6) * ((x1 * P) / Ftot)) / ...
            ((1 + 0.044 * exp(2390 / x6) * ((x2 * P) / Ftot) * (((x1 * P) / Ftot) ^ -0.5))^3));

end

%%
function frac = optimisationfileyield(optimvar , L, A_cross, T0, a, alpha, Ta, H, NA, Ap, Bp, Cp, Dp, Ep, Tcr,
     P, Fconv)
%optimvar(1)    ...FNH3
%optimvar(2)    ...ratio of CH4/NH3

    FCH4 = 0.95 * optimvar;
    Vspan = [0, L * A_cross];
    initial = [optimvar, FCH4, 0, 0, 0, T0];
    [~, x] = ode15s(@(V, x) func(V, x, a, alpha, Ta, H, NA, Ap, Bp, Cp, Dp, Ep, Tcr, P), Vspan, initial);
    F = x(:, 1:5);                          %fluxes in mol/s

%     frac = -(F(end, 3) / F(1, 2));                %minimize this fraction
%     frac = (((F(1, 2) - F(end, 2)) / F(1, 2)) - 1)^2;     %minimize the conversion
    frac = (1 / (F(end, 3) / F(1, 2)) - 1)^2;      %minimize this fraction
%     frac = Fconv - F(end, 3);                      %minimize this fraction

end
```

Listing 4: File used in section 3.1 to model the dependence on the inlet feed.

```
clear all;
close all;
clc;

option = optimoptions('fsolve','Display','off');

%Fallstudie 2 - NH3 Absorber

%% Heat Capacities
% A*t + B*t^2/2 + C*t^3/3 + D*t^4/4 + E/t + F - H (Shomate equation with t=
% T/1000)

cp_A_NH3 = 19.996;
```

```
cp_B_NH3 = 49.771;
cp_C_NH3 = -15.376;
cp_D_NH3 = 1.921;
cp_E_NH3 = 0.189;
cp_F_NH3 = -53.307;
cp_G_NH3 = 203.859;
cp_H_NH3 = -45.898;
cp_NH3 = [cp_A_NH3 cp_B_NH3 cp_C_NH3 cp_D_NH3 cp_E_NH3, cp_F_NH3, cp_G_NH3, cp_H_NH3];

cp_A_CH4 = -0.703;
cp_B_CH4 = 108.477;
cp_C_CH4 = -42.522;
cp_D_CH4 = 5.863;
cp_E_CH4 = 0.679;
cp_F_CH4 = -76.844;
cp_G_CH4 = 158.716;
cp_H_CH4 = -74.873;
cp_CH4 = [cp_A_CH4 cp_B_CH4 cp_C_CH4 cp_D_CH4 cp_E_CH4, cp_F_CH4, cp_G_CH4, cp_H_CH4];

cp_A_HCN = 32.694;
cp_B_HCN = 22.592;
cp_C_HCN = -4.369;
cp_D_HCN = -0.408;
cp_E_HCN = -0.282;
cp_F_HCN = 123.481;
cp_G_HCN = 233.260;
cp_H_HCN = 135.143;
cp_HCN = [cp_A_HCN cp_B_HCN cp_C_HCN cp_D_HCN cp_E_HCN, cp_F_HCN, cp_G_HCN, cp_H_HCN];

cp_A_H2SO4 = 47.289;
cp_B_H2SO4 = 190.331;
cp_C_H2SO4 = -148.130;
cp_D_H2SO4 = 43.866;
cp_E_H2SO4 = -0.74;
cp_F_H2SO4 = -758.952;
cp_G_H2SO4 = 301.296;
cp_H_H2SO4 = -735.130;
cp_H2SO4 = [cp_A_H2SO4 cp_B_H2SO4 cp_C_H2SO4 cp_D_H2SO4 cp_E_H2SO4, cp_F_H2SO4, cp_G_H2SO4, cp_H_H2SO4];

cp_A_H2 = 33.066;
cp_B_H2 = -11.363;
cp_C_H2 = 11.433;
cp_D_H2 = -2.773;
cp_E_H2 = -0.159;
cp_F_H2 = -9.981;
cp_G_H2 = 172.708;
cp_H_H2 = 0;
cp_H2 = [cp_A_H2 cp_B_H2 cp_C_H2 cp_D_H2 cp_E_H2, cp_F_H2, cp_G_H2, cp_H_H2];

cp_A_N2 = 28.986;
cp_B_N2 = 1.854;
cp_C_N2 = -9.647;
cp_D_N2 = 16.635;
cp_E_N2 = 0.000117;
cp_F_N2 = -8.672;
cp_G_N2 = 226.417;
cp_H_N2 = 0;
cp_N2 = [cp_A_N2 cp_B_N2 cp_C_N2 cp_D_N2 cp_E_N2, cp_F_N2, cp_G_N2, cp_H_N2];

% cp_A_H2O = 30.092;
% cp_B_H2O = 6.833;
% cp_C_H2O = 6.793;
% cp_D_H2O = -2.534;
% cp_E_H2O = 0.082;
% cp_F_H2O = -250.881;
% cp_G_H2O = 223.397;
% cp_H_H2O = -241.826;
% cp_H2O = [cp_A_H2O cp_B_H2O cp_C_H2O cp_D_H2O cp_E_H2O, cp_F_H2O, cp_G_H2O, cp_H_H2O];

cp_A_H2O = -203.606;
cp_B_H2O = 1523.29;
cp_C_H2O = -3196.413;
cp_D_H2O = 2474.455;
```

```
cp_E_H2O = 3.855;
cp_F_H2O = -256.54781;
cp_G_H2O = -488.716;
cp_H_H2O = -285.830;
cp_H2O = [cp_A_H2O cp_B_H2O cp_C_H2O cp_D_H2O cp_E_H2O, cp_F_H2O, cp_G_H2O, cp_H_H2O];

cp_ammsulf = 0; %Waermekapazitaet vernachlaessigbar

cp_G = [cp_NH3; cp_CH4; cp_HCN; cp_H2; cp_N2]; % parameters for gaseous components
cp_L = [cp_H2O; cp_H2SO4]; % parameters for liquid components

%% formation enthalpies (kJ/mol) at standard conditions (T=298K, P=1atm)

hf_NH3 = -45.94;
hf_CH4 = -74.6;
hf_HCN = 135.14;
hf_H2SO4 = -735.13; %gas
hf_H2O = -214.83;
hf_H2 = 0; %Element
hf_N2 = 0; %Element
hf_ammsulf = -1181;
hf_G = [hf_NH3; hf_CH4; hf_HCN; hf_H2; hf_N2]; % formation enthalpies of gaseous components
hf_L_in = [hf_H2O;hf_H2SO4]; % formation enthalpies of liquid components at inlet
hf_L_out = [hf_H2O; hf_H2SO4; hf_ammsulf]; % formation enthalpies of liquid components at outlet (with salt)
 %h_rxn = hf_ammsulf-hf_H2SO4-2*hf_NH3; %kJ/mol
%h_rxn = hf_ammsulf-2*hf_NH3-hf_H2SO4;
h_rxn=-111.9;
%% enthalpies at unknown temperature (Shomate)

% h_in1 = @(t) cp_G(:,1).*(t/1000) + (1/2).*cp_G(:,2).*(t/1000)^2 + (1/3).*cp_G(:,3).*(t/1000)^3 + (1/4).*cp_G
    (:,4).*(t/1000)^4 - cp_G(:,5)/(t/1000) + cp_G(:,6) - cp_G(:,7); %enthalpies in  of gas
% h_in2 = @(t) cp_L(:,1).*(t/1000)+ (1/2).*cp_L(:,2).*(t/1000)^2 + (1/3).*cp_L(:,3).*(t/1000)^3 + (1/4).*cp_L
    (:,4).*(t/1000)^4 - cp_L(:,5)/(t/1000) + cp_L(:,6) - cp_L(:,7);  % enthalpies in of liquid
% h_out1 = @(t) cp_G(:,1).*(t/1000) + (1/2).*cp_G(:,2).*(t/1000)^2 + (1/3).*cp_G(:,3).*(t/1000)^3 + (1/4).*cp_G
    (:,4).*(t/1000)^4 - cp_G(:,5)/(t/1000) + cp_G(:,6) - cp_G(:,7); % enthalpies out of gas
% h_out2 = @(t) [cp_L(:,1); 0].*(t/1000) + (1/2).*[cp_L(:,2); 0].*(t/1000)^2 + (1/3).*[cp_L(:,3); 0].*(t/1000)
    ^3 + (1/4).*[cp_L(:,4); 0].*(t/1000)^4 - [cp_L(:,5); 0]/(t/1000) + [cp_L(:,6); 0] - [cp_L(:,7); 0]; %
    enthalpies out of liquid

%considering only the first 4 parameters
h_in1 = @(t) cp_G(:,1).*(t/1000) + (1/2).*cp_G(:,2).*(t/1000)^2 + (1/3).*cp_G(:,3).*(t/1000)^3 + (1/4).*cp_G
    (:,4).*(t/1000)^4 ;%- cp_G(:,5)/(t/1000) ;%+ cp_G(:,6) - cp_G(:,7); %enthalpies in  of gas
h_in2 = @(t) cp_L(:,1).*(t/1000)+ (1/2).*cp_L(:,2).*(t/1000)^2 + (1/3).*cp_L(:,3).*(t/1000)^3 + (1/4).*cp_L
    (:,4).*(t/1000)^4 ;%- cp_L(:,5)/(t/1000) ;%+ cp_L(:,6) - cp_L(:,7);  % enthalpies in of liquid
h_out1 = @(t) cp_G(:,1).*(t/1000) + (1/2).*cp_G(:,2).*(t/1000)^2 + (1/3).*cp_G(:,3).*(t/1000)^3 + (1/4).*cp_G
    (:,4).*(t/1000)^4 ;%- cp_G(:,5)/(t/1000) ;%+ cp_G(:,6) - cp_G(:,7); % enthalpies out of gas
h_out2 = @(t) [cp_L(:,1); 0].*(t/1000) + (1/2).*[cp_L(:,2); 0].*(t/1000)^2 + (1/3).*[cp_L(:,3); 0].*(t/1000)^3
    + (1/4).*[cp_L(:,4); 0].*(t/1000)^4 ;%- [cp_L(:,5); 0]/(t/1000) ;%+ [cp_L(:,6); 0] - [cp_L(:,7); 0]; %
    enthalpies out of liquid


%% Flow Data non-ideal (Ramona)
%FluxReactor = [0.72, 6.8, 20, 74, 2.8]; %mol/s (coming from reactor)
FluxReactor = [1.56, 5.9, 15.4, 54.3, 2.73];

%Liquid & Gas Inlet flows [mol/s]
L_in_H2SO4 = FluxReactor(1)*0.5*1.4; % molar flow H2SO4 (10% H2SO4 in H2O) [mol/s]
L_in = L_in_H2SO4*10; % total molar flow (liquid) [mol/s]
L_in_H2O = L_in-L_in_H2SO4; % molar flow H2O [mol/s]
G_in = sum(FluxReactor);  %total molar flow (gas) same for ideal & non-ideal [mol/s]

%% Flow Data ideal
%Liquid Inlet flows
L_in_id = L_in; %[mol/s]
L_in_H2SO4_id = L_in_H2SO4; %[mol/s]
L_in_H2O_id = L_in_id - L_in_H2SO4_id; %[mol/s]


%molar fractions of inlet (x:gas, y:liquid)
x_NH3_in = FluxReactor(1)/sum(FluxReactor);
x_CH4_in = FluxReactor(2)/sum(FluxReactor);
x_HCN_in = FluxReactor(3)/sum(FluxReactor);
x_H2_in = FluxReactor(4)/sum(FluxReactor);
```

```
x_N2_in = FluxReactor(5)/sum(FluxReactor);


y_H2O_in = 0.9;
y_H2SO4_in = 0.1;
x_in1 = [x_NH3_in; x_CH4_in; x_HCN_in; x_H2_in; x_N2_in]; % molar fractions  of gas in
x_in2 = [y_H2O_in; y_H2SO4_in]; % molar fractions of liquid in

%Gas outlet flow
G_out = sum(FluxReactor)-(FluxReactor(1)-10^(-4)*sum(FluxReactor)); % [mol/s]
%molar fractions of outlet
x_CH4_out = FluxReactor(2)/G_out;
x_HCN_out = FluxReactor(3)/G_out;
x_H2_out = FluxReactor(4)/G_out;
x_N2_out = FluxReactor(5)/G_out;
x_NH3_out = 10^(-4);

G_NH3_in=x_NH3_in*G_in;  %[mol/s]
G_NH3_out=x_NH3_out*G_out; %[mol/s]
delta_G_NH3 = G_NH3_in-G_NH3_out; %[mol/s]

%Liquid outlet flow (nonideal & ideal)
L_out=L_in; %[mol/s]
L_out_id = L_in_id; %[mol/s]

%molar fractions outlet
y_H2SO4_out = (L_in_H2SO4-0.5*delta_G_NH3)/L_out;
y_H2O_out = L_in_H2O/L_out;
y_ammsulf_out = 0.5*delta_G_NH3/L_out; %0.5 because 2 NH3 --> 1 ammonium sulfate
x_out1 = [x_NH3_out; x_CH4_out; x_HCN_out; x_H2_out; x_N2_out]; % molar fractions of gas out
x_out2 = [y_H2O_out; y_H2SO4_out; y_ammsulf_out]; % molar fractions of liquid out

%Temperature of inlet gas stream
t_in_G = 90+273.15; %K
%Temperature of inlet liquid stream
t_in_L = 20+273.15; % K

t0 = 60+273; %K
H_in1 = @(t) sum(x_in1 .* h_in1(t_in_G)); % gas enthalpies in
H_out1 = @(t) sum(x_out1 .* h_out1(t)); % gas enthalpies out
H_in2 = @(t) sum(x_in2 .* h_in2(t_in_L)); % liquid enthalpies in
H_out2 = @(t) sum(x_out2 .* h_out2(t)); % liquid enthalpies out

%% Energy balance: non-ideal

EB = @(t) G_in.*H_in1(t) + L_in.*H_in2(t) - G_out.*H_out1(t) - L_out.*H_out2(t) + h_rxn.*(G_NH3_in-G_NH3_out);

t_opt = fsolve(EB,t0, option);
T_ab1_nonid = t_opt-273.15; %Celsius
fprintf('Nonideal temperature: T= %g\n', T_ab1_nonid);

%% Energy balance: ideal
h_in_id1 = @(tt) cp_G(:,1).*(tt/1000); %gas enthalpies in
h_in_id2 = @(tt) cp_L(:,1).*(tt/1000); %liquid enthalpies in
h_out_id1 = h_in_id1; % gas enthalpies out
h_out_id2 = @(tt) [cp_L(:,1); 0].*(tt/1000); %liquid enthalpies out

H_in_id1 = @(tt) sum(x_in1 .* h_in_id1(t_in_G)); % gas enthalpies in
H_out_id1 = @(tt) sum(x_out1 .* h_out_id1(tt)); % gas enthalpies out
H_in_id2 = @(tt) sum(x_in2 .* h_in_id2(t_in_L)); % liquid enthalpies in
H_out_id2 = @(tt) sum(x_out2 .* h_out_id2(tt)); % liquid enthalpies out

EB_id = @(tt) G_in.*H_in_id1(tt) + L_in_id.*H_in_id2(tt) - G_out.*H_out_id1(tt) - L_out_id.*H_out_id2(tt) +
     h_rxn.*(G_NH3_in-G_NH3_out);
t_opt_id = fsolve(EB_id,t0, option);
T_ab1_id = t_opt_id - 273.15; %Celsius
fprintf('ideal temperature: T= %g\n', T_ab1_id)

%% Dimensions non-ideal case
D = 1.3; % m (Diameter)
A_c = pi/4 * D^2; % m^2
P = 1.013; % bar
P2 = 101325; %Pa
```

```
NTU = log(x_NH3_in/x_NH3_out); % gas
NTU = 6;

% Calculation of KG*a
R = 8.314; %J/(mol*K) J=f(kg, m, s)
a = 190; %m^2/m^3
ap=492; %1/m
Q = @(t)G_in*1000*R*t / (P2); %m^3/s assuming ideal gas
v= @(t) Q(t)/A_c; %m/s
M_H2 = 2.016; %g/mol
M_NH3 = 17.031; %g/mol
M_CH4 = 16; %g/mol
M_N2 = 28; %g/mol
M_HCN = 27; %g/mol
Vm_H2 = 2*1.98; %from Cussler (in Angstrom)
Vm_NH3 = 19.5+3*1.98; %from Cussler
Diff = @(t) 10^(-3) * (t^1.75 * sqrt((1/M_H2)+(1/M_NH3)))/(((Vm_H2^(1/3)+Vm_NH3^1/3))^2)*10^(-4); % m^2/s
d= 0.025; %m
%viscosity
TH2 = [0, 20, 50, 100, 200, 300, 400, 500, 600] + 273.15;               %temperature in K (from
    engineeringtoolbox)
nuH2 = [0.84, 0.88, 0.94, 1.04, 1.21, 1.37, 1.53, 1.69, 1.84] .* 1e-5;  %in Pa*s (dynamic viscosity)
mdlH2 = polyfit(TH2, nuH2, 2); %polynomial fit
bH2 = mdlH2(1);
mH2 = mdlH2(2);
uH2 = mdlH2(3);
muH2 = @(t) bH2 .* (t .^2) + mH2 .* t + uH2; %Pa*s

G_m = G_in*(x_H2_in*M_H2 + x_NH3_in*M_NH3 + x_CH4_in*M_CH4+x_HCN_in*M_HCN)/1000; %kg/s
kG= @(t) 5.23 * A_c * ap * Diff(t)/(R*t)*(G_m/(a*muH2(t)))^.7*(muH2(t)/(P2*M_H2/(R*t))/Diff(t))^(1/3)*(d*ap)
    ^(-2)*P2; %overall mass transfer coefficient (mol/h/m^2/bar)
% kG = @(t) kG./(R.*t);
HTU = @(t) G_in*1000/3600/ ( A_c * kG(t) * a); %m NOTE: the higher the temperature, the lower HTU
H = @(t) HTU(t) * NTU + 0.5; % m

Vol = @(t) A_c*H(t); %m^3

%% Dimensions ideal case
Q_id= @(tt)G_in*1000*R*tt / (P2); %m^3/s
v_id= @(tt) Q(tt)/A_c; %m/s
Diff_id = @(tt) 10^(-3) * (tt^1.75 * sqrt((1/M_H2)+(1/M_NH3)))/(((Vm_H2^(1/3)+Vm_NH3^1/3))^2)*10^(-4); % m^2/s
muH2_id = @(tt) bH2 .* (tt .^2) + mH2 .* tt + uH2; %Pa*s
kG_id= @(tt) 5.23 * A_c * ap * Diff(tt)/(R*tt)*(G_m/3600/(ap*muH2(tt)))^.7*(muH2(tt)/(P2*M_H2/(R*tt))/Diff(tt))
    ^(1/3)*(d*ap)^(-2)*P2; %overall mass transfer coefficient (mol/h/m^2/bar)
HTU_id = @(tt) G_in/ ( A_c * kG(tt) * a); %m
H_id = @(tt) HTU_id(tt) * NTU + 0.5; % m
Vol_id = @(tt) A_c*H_id(tt); %m^3

%% Costs for non-ideal case

%Pricesc
p_H2SO4 = 75; %US-$ per ton
p_coolingwater = 0.1; %US-$ per ton
p_chilledwater = 0.2; %US-$ per ton
p_waste = 2; %US-$ per ton

%Costs of the medium depend on the liquid inlet flow!!!

%molar weights
M_H2SO4 = 98.079; %g/mol
M_ammsulf = 134.14; %g/mol
M_H2O = 18.01528; %g/mol

%masses per hour
m_H2SO4 = L_in_H2SO4 * M_H2SO4 /1000000*3600; %ton/h (mol/s * g/mol /1000000*3600)
m_H2O = L_in_H2O * M_H2O / 1000000*3600; %ton/h
m_waste =  L_in * (y_H2O_out*M_H2O + y_H2SO4_out * M_H2SO4 + y_ammsulf_out * M_ammsulf) /1000000*3600;
%masses per year
m_H2SO4_y = m_H2SO4 * 8000; %ton/year
m_H2O_y = m_H2O * 8000; % ton/year
m_waste_y = m_waste * 8000; %ton/year

% costs per hour
```

```
c_H2SO4 = p_H2SO4 * m_H2SO4; % US-$ per h
c_H2O = p_coolingwater * m_H2O; % US-$ per h
c_waste = p_waste * m_waste; %US-$ per h

%costs per year
c_H2SO4_y = c_H2SO4 * 8000; %US-$ per year
c_H2O_y = c_H2O * 8000; %US-$ per year
c_waste_y = c_waste * 8000; %US-$ peryear

%column price depends on the HEIGHT, which depends on the gas flow (same
%for both cases)
c_column = @(t) 70000*(H(t)*D)^.5; %US-$

%TOTAL costs over 10 years
c_tot = c_column(t_opt) + (c_H2SO4_y+c_H2O_y+c_waste_y)*10; % US-$ over 10years
fprintf('Gesamtkosten nicht ideal:  %g\n', c_tot)
%% Costs for ideal case

%masses per hour
m_H2SO4_id = L_in_H2SO4_id * M_H2SO4 /1000000*3600; %ton/h
m_H2O_id = L_in_H2O_id * M_H2O /1000000*3600; %ton/h
m_waste_id = L_in_id * (y_H2O_out*M_H2O + y_H2SO4_out * M_H2SO4 + y_ammsulf_out * M_ammsulf) /1000000*3600; %
    ton/h
c_column_id = @(tt) 70000*(H_id(tt)*D)^.5;
%masses per year
m_H2SO4_id_y = m_H2SO4_id * 8000; %ton/year
m_H2O_id_y = m_H2O_id * 8000; %ton/year
m_waste_id_y = m_waste_id * 8000; %ton/year

%costs per hour
c_H2SO4_id = p_H2SO4 * m_H2SO4_id; % US-$ per h
c_H2O_id = p_coolingwater * m_H2O_id; % US-$ per h
c_waste_id = p_waste * m_waste_id; %US-$ per h

%costs per year
c_H2SO4_id_y = c_H2SO4_id * 8000; %US-$ per year
c_H2O_id_y = c_H2O_id * 8000; %US-$ per year
c_waste_id_y = c_waste_id * 8000; %US-$ per year

c_column_id = @(tt) 70000*(H_id(tt)*D)^.5; %US-$

%TOTAL costs over 10 years
c_tot_id = c_column_id(t_opt_id)+(c_H2SO4_id_y + c_H2O_id_y + c_waste_id_y)*10; %US-$ over 10 years
fprintf('Gesamtkosten ideal:  %g\n', c_tot_id)
%% Absorber 2
% Gas inlet: NH3, CH4, H2, N2, HCN --> x_out1 (from absorber 1)
% Liq inlet: H2O
% Gas outlet: NH3, CH4, H2, N2, HCN (100ppm)
% Liq outlet: H2O, HCN

% G_i_in = x_out1.*G_out; %G_i_in in 2. Absorber
% G_out_ab2 = G_out-G_i_in(3);
% y_in_ab2 = 1; %H2O pure
% delta_HCN = (x_out1(3) - 10^-4);
% delta_HCN4 = delta_HCN / 4;
% x_out_ab2 = [x_NH3_out+delta_HCN4; x_CH4_out+delta_HCN4; 10^-4; x_H2_out+delta_HCN4; x_N2_out+delta_HCN4];
% y_out_ab2 = [1-delta_HCN; delta_HCN];
% %assume: G_in = G_out and L_in = L_out
% G_in_ab2=G_out;
% G_out_ab2 = G_in_ab2 - x_out1(3) + x_out_ab2(3); %kmol/h
% L_out_ab2 = L_in_ab2+delta_HCN*G_in_ab2; %kmol/h
%
% %% Non-ideal Energy Balance
% %enthalpy difference of water and HCN in liquid stream
% dh_in_H2O_L = @(t2) cp_L(1,1).*(t2/1000) + (1/2).*cp_L(1,2).*(t2/1000)^2 + (1/3).*cp_L(1,3).*(t2/1000)^3 +
    (1/4).*cp_L(1,4).*(t2/1000)^4 - cp_L(1,5)/(t2/1000) + cp_L(1,6) - cp_L(1,7); % enthalpies out of gas
% dh_in_HCN_G =  @(t2) cp_G(3,1).*(t2/1000) + (1/2).*cp_G(3,2).*(t2/1000)^2 + (1/3).*cp_G(3,3).*(t2/1000)^3 +
    (1/4).*cp_G(3,4).*(t2/1000)^4 - cp_G(3,5)/(t2/1000) + cp_G(3,6) - cp_G(3,7); % enthalpies out of gas
% c
% h_in_ab2_L = cp_L(1,1).*(293.15/1000) + (1/2).*cp_L(1,2).*(293.15/1000)^2 + (1/3).*cp_L(1,3).*(293.15/1000)^3
     + (1/4).*cp_L(1,4).*(293.15/1000)^4 - cp_L(1,5)/(293.15/1000) + cp_L(1,6) - cp_L(1,7);; %enthalpy of
    liquid flow in
% h_out_ab2_L = @(t2) [dh_in_H2O_L(t2); dh_in_HCN_G(t2)]; %enthalpy of liquid flow out
```

```
% h_in_ab2_G = h_in1(t_opt); %enthalpy of gas flow in
% h_out_ab2_G = @(t2) h_out1(t2); % enthalpy of gas flow out
%
% H_in_ab2_L = y_in_ab2*h_in_ab2_L;
% H_in_ab2_G = sum(x_in_ab2 .* h_in_ab2_G);
% H_out_ab2_L = @(t2) sum(y_out_ab2 .* h_out_ab2_L(t2));
% H_out_ab2_G = @(t2) sum(x_out_ab2 .* h_out_ab2_G(t2));
%
% EB_ab2 = @(t2) G_in_ab2.*H_in_ab2_G-G_out_ab2.*H_out_ab2_G(t2)+L_in_ab2.*H_in_ab2_L-L_out_ab2.*H_out_ab2_L(t2
    );
% t_opt_ab2 =olve(EB_ab2,300);
% T_ab2_nonid = t_opt_ab2 - 273.15;
%
% %% Ideal Energy Balance
% h_in_ab2_id = h_in_id1(t_opt_id); %enthalpy inlet gas
% H_in_ab2_id = sum(x_in_ab2 .* h_in_ab2_id); % total enthalpy inlet gas
% h_out_ab2_id = @(tt2) h_out_id1(tt2); % enthalpy outlet gas
% H_out_ab2_id = @(tt2) sum(x_out_ab2 .* h_out_ab2_id(tt2)) %total enthalpy outlet gas
% H_in_ab2_id_L = cp_L(1,1).*(293.15/1000);
% h_out_ab2_id_L = @(tt2) [cp_L(1,1).*(tt2/1000); cp_G(3,1).*(tt2/1000)];
% H_out_ab2_id_L = @(tt2)sum(y_out_ab2 .* h_out_ab2_id_L(tt2));
%
% EB_ab2_id = @(tt2) G_in_ab2*H_in_ab2_id-G_out_ab2*H_out_ab2_id(tt2)+L_in_ab2*H_in_ab2_id_L- L_in_ab2*
    H_out_ab2_id_L(tt2);
% t_opt_ab2_id = fsolve(EB_ab2_id,300);
% T_ab2_id = t_opt_ab2_id - 273.15;
%
 %% Dimensions non-ideal case
% D = 1.3; % m (Diameter)
% A_c = pi/4 * D^2; % m^2
% P = 1.013; % bar
% P2 = 101325; %Pa
%
% % Calculation of KG*a
% M_H2O = 18.0153;
% M_HCN = 27.025;
% R = 8.314; %J/(mol*K) J=f(kg, m, s)
% a = 190; %m^2/m^3
% Q_ab2= @(t2)G_in_ab2*1000*R*t2 / P2; %m^3/h assuming ideal gas
% v_ab2= @(t2) Q(t2)/A_c/3600; %m/s
% Vm_HCN = 1.98+16.5+5.69; %from Cussler (in Angstrom)
% Vm_NH3 = 2*1.98+5.48; %from Cussler
% d= 0.025; %m
%
% %viscosity
% TH2O = [0, 5, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100] + 273.15;              %temperature in K (from
    engineeringtoolbox)
% nuH2O = [1.787, 1.519, 1.307, 1.002, 0.798, 0.653, 0.547, 0.467, 0.404, 0.355, 0.315, 0.282] .* 1e-3;  %in Pa
    *s (dynamic viscosity)
% mdlH2O = polyfit(TH2, nuH2, 2); %polynomial fit
% bH2O = mdlH2O(1);
% mH2O = mdlH2O(2);
% uH2O = mdlH2O(3);
% muH2O = @(t) bH2O .* (t .^2) + mH2O .* t + uH2O; %nu = mu/rho (%cm^2/s)
% Diff2_l = @(t2) 1.173 * 10^-3*sqrt(2.6*M_H2O)*t2/(muH2O(t2)*1000*Vm_HCN^.6); % m^2/s
% Diff2_g = @(t2) 10^(-3) * (t2^1.75 * sqrt((1/M_H2)+(1/M_HCN)))/(((Vm_H2^(1/3)+Vm_HCN^1/3))^2)*10^(-4); % m^2/
    s
%
% G_m = G_in*(x_H2_in*M_H2 + x_NH3_in*M_NH3 + x_CH4_in*M_CH4+x_HCN_in*M_HCN); %kg/h
% kG= @(t) 5.23 * A_c * a * Diff(t)/(R*t)*(G_m/3600/(a*muH2(t)))^.7*(muH2(t)/(P2*M_H2/(R*t))/Diff(t))^(1/3)*(d*
    a)^(-2)*P2; %overall mass transfer coefficient (mol/h/m^2/bar)
% HTU = @(t) G_in*1000/3600/ ( A_c * kG(t) * a); %m NOTE: the higher the temperature, the lower HTU
% H = @(t) HTU(t) * NTU + 0.5; % m
% Vol = @(t) A_c*H(t); %m^3
```

Listing 5: File used in section 3.2 to model the first absorber.

```
function absorber2
clear; close all; clc;

% FluxReactor = [0.72, 6.8, 20, 74, 2.8]; %mol/s from reactor
FluxReactor = [1.56, 5.9, 15.4, 54.3, 2.73];
```

```
Tin_G = [90 + 273.15, 90 + 273.15]; %Inlet temperature from first absorber: (1):nonideal, (2):ideal

x_NH3_in1 = FluxReactor(1)/sum(FluxReactor);
x_CH4_in1 = FluxReactor(2)/sum(FluxReactor);
x_HCN_in1 = FluxReactor(3)/sum(FluxReactor);
x_H2_in1 = FluxReactor(4)/sum(FluxReactor);
x_N2_in1 = FluxReactor(5)/sum(FluxReactor);

x_in_abs1 = [ x_NH3_in1, x_CH4_in1, x_HCN_in1, x_H2_in1, x_N2_in1];

G_out1 = sum(FluxReactor)-(0.72-10^(-4));
x_CH4_out1 = FluxReactor(2)/G_out1;
x_HCN_out1 = FluxReactor(3)/G_out1;
x_H2_out1 = FluxReactor(4)/G_out1;
x_N2_out1 = FluxReactor(5)/G_out1;
x_NH3_out1 = 10^(-4);

G_in2= G_out1;
x_CH4_in2 = 5.9/G_out1;
x_HCN_in2 = 15.4/G_out1;
x_H2_in2 = 54.3/G_out1;
x_N2_in2 = 2.73/G_out1;
x_NH3_in2 = 10^(-4);

x_in_abs2 = [ x_NH3_in2, x_CH4_in2, x_HCN_in2, x_H2_in2, x_N2_in2];

G_out_abs2 = G_in2 - (x_HCN_in2-10^-4)*G_in2;
x_HCN_out2 = 10^(-4);
x_CH4_out2 = FluxReactor(2)/G_out_abs2;
x_H2_out2 = FluxReactor(4)/G_out_abs2;
x_N2_out2 = FluxReactor(5)/G_out_abs2;
x_NH3_out2 = (10^(-4)*G_in2)/G_out_abs2;

x_out_abs2 = [x_NH3_out2, x_CH4_out2, x_HCN_out2, x_H2_out2, x_N2_out2];

G_i_in_abs2 = G_in2.*x_in_abs2;
G_i_out_abs2 = G_out_abs2.*x_out_abs2;
L_in_abs2 = 1500e3 / 3600; % angenommen
y_H2O_abs2_in = 1;
y_HCN_abs2_in = 0;
L_i_in_abs2 = [L_in_abs2 * y_H2O_abs2_in; L_in_abs2 * y_HCN_abs2_in];

G_HCNabs = G_i_in_abs2(3)-G_i_out_abs2(3);

L_out_abs2 = G_HCNabs + L_in_abs2;
y_H2O_abs2_out = L_in_abs2/L_out_abs2;
y_HCN_abs2_out = 1 - y_H2O_abs2_out;
y_abs2_out = [y_H2O_abs2_out, y_HCN_abs2_out];
fprintf('x_abs2_out: %g\n', y_abs2_out);
L_i_out_abs2 = [L_out_abs2 * y_H2O_abs2_out; L_out_abs2 * y_HCN_abs2_out];
fprintf('L_i_out_abs2: %g\n', L_i_out_abs2);

%Energy balance for the non-ideal case
guess = 25 + 273.15;
option = optimoptions('fsolve','Display','off');
T_nonideal = fsolve(EBfuncNonideal(G_i_in_abs2, G_i_out_abs2, L_i_in_abs2, L_i_out_abs2, Tin_G(1)), guess,
    option);
fprintf('Non ideal temperature: T= %g\n', T_nonideal-273.15);

%Energy balance for the ideal case
guess = 25 + 273.15;
T_ideal = fsolve(EBfuncIdeal(G_i_in_abs2, G_i_out_abs2, L_i_in_abs2, L_i_out_abs2, Tin_G(2)), guess, option);
fprintf('Ideal temperature: T= %g\n', T_ideal-273.15);

%Calculation of NTU
NTU = NTUfunc(T_nonideal, T_ideal, y_HCN_abs2_out, x_HCN_in2, x_HCN_out2);
fprintf('NTU non ideal: %g \t NTU ideal: %g \n', NTU);

%Calculation of HTU
d = 2;                % gesetzt Durchmesser 1m
Ak = ((d^2)/4)*pi;      %crossectional area (variabler Wert)
[HG] = HGfunc(Ak, T_nonideal, T_ideal, G_in2);
```

```
%Calculation of HL
HTU = HLfunc(T_nonideal, T_ideal, L_in_abs2, Ak, G_in2, HG);
fprintf('HTU non ideal: %g \t HTU ideal: %g \n', HTU);

% Calculate height of absorber 2
H_reactor = HTU.*NTU+0.5;
fprintf('height reactor non ideal: %g \t ideal: %g \n', H_reactor);

end

%%
function EBfuncNonideal = EBfuncNonideal(G_in, G_out, L_in, L_out, Tin_G)

Tin = [Tin_G, 293.15]; %(1): Gas temperature, (2) liquid temperature (fix)
[cp_G, cp_L] = HeatcapacityParameter();

h_G = @(t) cp_G(:,1).*(t/1000) + (1/2).*cp_G(:,2).*((t/1000)^2); %+ (1/3).*cp_G(:,3).*((t/1000)^3) + (1/4).*
     cp_G(:,4).*((t/1000)^4) - cp_G(:,5)./(t/1000) + cp_G(:,6) - cp_G(:,7); %enthalpies in  of gas
h_L = @(t) cp_L(:,1).*(t/1000) + (1/2).*cp_L(:,2).*(t/1000)^2;% + (1/3).*cp_L(:,3).*(t/1000)^3 + (1/4).*cp_L
     (:,4).*(t/1000)^4 - cp_L(:,5)./(t/1000) + cp_L(:,6) - cp_L(:,7); % enthalpies in of liquid

EBfuncNonideal = @(t)   sum(G_in' .* h_G(Tin(1))) + sum(L_in .* h_L(Tin(2))) - ...
                 sum(G_out' .* h_G(t)) - sum(L_out .* h_L(t));
end

%%
function EBfuncIdeal = EBfuncIdeal(G_in, G_out, L_in, L_out, Tin_G)

Tin = [Tin_G, 293.15]; %(1): Gas temperature, (2) liquid temperature (fix)
[cp_G, cp_L] = HeatcapacityParameter();

h_G = @(t) cp_G(:,1).*(t/1000); %enthalpies in  of gas
h_L = @(t) cp_L(:,1).*(t/1000); % enthalpies in of liquid

EBfuncIdeal = @(t)   sum(G_in' .* h_G(Tin(1))) + sum(L_in .* h_L(Tin(2))) - ...
                 sum(G_out' .* h_G(t)) - sum(L_out .* h_L(t));
end

%%
function [cp_G, cp_L] = HeatcapacityParameter()
% A*t + B*t^2/2 + C*t^3/3 + D*t^4/4 + E/t + F - H (Shomate equation with t=
% T/1000)

cp_A_NH3 = 19.996;
cp_B_NH3 = 49.771;
cp_C_NH3 = -15.376;
cp_D_NH3 = 1.921;
cp_E_NH3 = 0.189;
cp_F_NH3 = -53.307;
cp_G_NH3 = 203.859;
cp_H_NH3 = -45.898;
cp_NH3 = [cp_A_NH3 cp_B_NH3 cp_C_NH3 cp_D_NH3 cp_E_NH3, cp_F_NH3, cp_G_NH3, cp_H_NH3];

cp_A_CH4 = -0.703;
cp_B_CH4 = 108.477;
cp_C_CH4 = -42.522;
cp_D_CH4 = 5.863;
cp_E_CH4 = 0.679;
cp_F_CH4 = -76.844;
cp_G_CH4 = 158.716;
cp_H_CH4 = -74.873;
cp_CH4 = [cp_A_CH4 cp_B_CH4 cp_C_CH4 cp_D_CH4 cp_E_CH4, cp_F_CH4, cp_G_CH4, cp_H_CH4];

cp_A_HCN = 32.694;
cp_B_HCN = 22.592;
cp_C_HCN = -4.369;
cp_D_HCN = -0.408;
cp_E_HCN = -0.282;
cp_F_HCN = 123.481;
cp_G_HCN = 233.260;
cp_H_HCN = 135.143;
cp_HCN = [cp_A_HCN cp_B_HCN cp_C_HCN cp_D_HCN cp_E_HCN, cp_F_HCN, cp_G_HCN, cp_H_HCN];
```

```
cp_A_H2 = 33.066;
cp_B_H2 = -11.363;
cp_C_H2 = 11.433;
cp_D_H2 = -2.773;
cp_E_H2 = -0.159;
cp_F_H2 = -9.981;
cp_G_H2 = 172.708;
cp_H_H2 = 0;
cp_H2 = [cp_A_H2 cp_B_H2 cp_C_H2 cp_D_H2 cp_E_H2, cp_F_H2, cp_G_H2, cp_H_H2];

cp_A_N2 = 28.986;
cp_B_N2 = 1.854;
cp_C_N2 = -9.647;
cp_D_N2 = 16.635;
cp_E_N2 = 0.000117;
cp_F_N2 = -8.672;
cp_G_N2 = 226.417;
cp_H_N2 = 0;
cp_N2 = [cp_A_N2 cp_B_N2 cp_C_N2 cp_D_N2 cp_E_N2, cp_F_N2, cp_G_N2, cp_H_N2];

cp_A_H2O = -203.606;
cp_B_H2O = 1523.29;
cp_C_H2O = -3196.413;
cp_D_H2O = 2474.455;
cp_E_H2O = 3.855;
cp_F_H2O = -256.54781;
cp_G_H2O = -488.716;
cp_H_H2O = -285.830;
cp_H2O = [cp_A_H2O cp_B_H2O cp_C_H2O cp_D_H2O cp_E_H2O, cp_F_H2O, cp_G_H2O, cp_H_H2O];

cp_G = [cp_NH3; cp_CH4; cp_HCN; cp_H2; cp_N2]; % parameters for gaseous components
cp_L = [cp_H2O; cp_HCN]; % parameters for liquid components

end
%%
function [NTU] =  NTUfunc(T_nonideal, T_ideal, y_HCN_abs2_out, x_HCN_abs2_in, x_HCN_out2)

T = [T_nonideal, T_ideal];
T0 = [(1/298.15), (1/298.15)];
% Define constants
x_i_out = [y_HCN_abs2_out, y_HCN_abs2_out];
y_i_in = [x_HCN_abs2_in, x_HCN_abs2_in];
y_i_out = [x_HCN_out2, x_HCN_out2];
y_i_out_G = [0, 0];

% Calculate Henry coef (Quelle: webbook nist)
% Henry = 6*10^5;  %Henry coeff [Pa] PPP 12*5000*((1/T)-(1/298.15));
Henry = (12.*5000.*((1./T)-T0))*0.02703;
y_i_in_G = Henry./(10^5).*x_i_out;

NTU = (y_i_in-y_i_out)./(((y_i_in-y_i_in_G)-(y_i_out-y_i_out_G))./(log((y_i_in-y_i_in_G)./(y_i_out-y_i_out_G)))
    );

end
%%
function [HG] = HGfunc(Ak, T_nonideal, T_ideal, G_in)

% kG = 114*1000*(1/3600)*(1/(10^5));

% define constants
T = [T_nonideal, T_ideal];  % Temperatur in [K]
a = 190;                     % packing are per bed volume [m^2/m^3]
p = 10^5;                    % Druck im Absorber
R = 8.3144;                  % Gaskonstante
MW_H2 = 0.002;
Q = (G_in.*R.*T)./p;         % Flussgeschwindigkeit
v = Q/Ak;                   % Geschwindigkeit [m/s]
Gm = G_in*MW_H2;            % [kg/s] Gasfluss
rhoH2 = (p.*MW_H2)./(R.*T); % ideales Gas
d = 0.025;                  % [m]

% viscosity
TH2 = [0, 20, 50, 100, 200, 300, 400, 500, 600] + 273.15;              %temperature in K (from
```

```
    engineeringtoolbox)
nuH2 = [0.84, 0.88, 0.94, 1.04, 1.21, 1.37, 1.53, 1.69, 1.84] .* 1e-5;  %in Pa*s (dynamic viscosity)
mdlH2 = polyfit(TH2, nuH2, 2); %polynomial fit
bH2 = mdlH2(1);
mH2 = mdlH2(2);
uH2 = mdlH2(3);
muH2 = bH2 .* (T .^2) + mH2 .* T + uH2;

nuH2 = muH2./rhoH2;

% Calculate diffusion coefficient
M_HCN = 27.025; %[g/mo]
Vm_HCN = 1.98+16.5+5.69; %from Cussler (in Angstrom)
M_H2 = 2.016; %g/mol
Vm_H2 = 2*1.98; %from Cussler (in Angstrom)

D = (10^(-3) * (T.^1.75 .* sqrt((1./M_H2)+(1./M_HCN))))./(((Vm_H2.^(1/3)+Vm_HCN.^1/3))^2))*10^(-4); % m^2/s

% Calculate Schmidt number
Sc = nuH2./D;

% calculate kG (diffusion HCN in H2) also Rueckwaerts
% kG = 5.23.*ad.*(p./(R.*T)).*((Gm./(ad.*muH2)).^(0.7)).*(Sc.^(1/3)).*((ad.*d).^(-2)).*p; % [cm/s]

kG = 3.6*((v/(a*nuH2))^(0.7))*((nuH2/D)^(1/3))*((a*d)^(-2))*(a*D);
kG = kG./(R.*T);

% calculate HG
HG = (G_in./(Ak.*1.013.*10^5.*kG.*a));

end
%%
function [HTU] = HLfunc(T_nonideal, T_ideal, Lm, Ak, Gm, HG)

T = [T_nonideal, T_ideal];

rho = 1000;  % density of water in kg/m^3
%viscosity of water
TH2O = [0, 5, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100] + 273.15;               %temperature in K (from
    engineeringtoolbox)
muH2O = [1.787, 1.519, 1.307, 1.002, 0.798, 0.653, 0.547, 0.467, 0.404, 0.355, 0.315, 0.282] .* 1e-3;  %in Pa*s
     (dynamic viscosity)
mdlH2O = polyfit(TH2O, muH2O, 2); %polynomial fit
bH2O = mdlH2O(1);
mH2O = mdlH2O(2);
uH2O = mdlH2O(3);
muH2O = (bH2O .* (T .^2) + mH2O .* T + uH2O); %Pa*s
g = 9.81; % [m/s^2]
MWH2O = 0.018;
LG = Lm/Gm;

a = 190; % total surface [m^2/m^3] (gegeben in Aufgabenstellung)
ad = 492;  % packing factor [1/m pack] (dry)
aw = 492;     % packing factor (wet)

% calculation of diffusion coefficient in liquid
D = 4.4*10^(-8).*(T./(muH2O*1000)); %[cm^2/s]
D = D/(10^4); %[m/s]
d = 0.025; % nominal packing size [m]
Sc = muH2O/(rho*D);

% calculation of kL
% kL = (rho./(muH2O.*g)).^(-1/3).*0.0051.*((Lm./(aw.*muH2O)).^(2/3)).*Sc^(-1/2).*(ad.*d).^0.4.*(MWH2O./rho);
v0 = (18.06/(10^6)*Lm)/Ak; % 18.06 molare Volumen Wasser
kL = 0.0051.*((v0./(a.*muH2O)).^(0.67)).*((D./muH2O).^(0.5)).*((a.*d).^0.4).*((1./(muH2O.*g)).^(-1/3));
kL = kL .* rho ./ 0.018;
% calculation of HL
Lm = Lm*0.018;  % liquid stream [kg/s]
HL = (Lm./(Ak.*kL.*a));


T0 = [(1/298.15), (1/298.15)];
Henry = (12.*5000.*((1./T)-T0))*0.02703;
```

```
m = Henry/(10^5);
% m=6;

HTU = HG + (m./LG).*HL;

end
```

Listing 6: File used in section 3.3 to model the second absorber.

```
function [totex, capex, opex, D, N_stages, Height, Diam, B, RR, RR_min] = mccabe_raoultfunc(z1, P_atm, q,
    R_param, x_D, x_B, F)
%% Distillation column
% This function calculates the dimensions of a distillation column using
% the Mccabe-Thiele method based on the ideal mixture (raoults law) model for VLE
% The input paramaters are as follows:
% P = Pressure [atm]
% q = feed quality
% R_param = multiplying factor of reflx ratio

% Tin = Temperature at which feed is delivered [C]
% z1 = molar fraction of HCN in feed
% x_D = Fraction of HCN in distillate
% x_B = fraction of HCN in bottoms
% F = feed rate [mol/h];

%Convert pressure from atmospheres to mmHg
P = P_atm * 760;

%Calculate the equilibrium line using raoult VLE
x1 = linspace(0.0000001,1,1000);  %Calculate the equilibrium line using raoult VLE
y1 = zeros(0,length(x1));
Tmix = zeros(0, length(x1));

for g = 1:length(x1)
[y1(g), Tmix(g)] = raoult(x1(g),P);
end

%% Mass Balances

D = F * (z1 - x_B) / (x_D - x_B); %Calculate distillate rate [mol/h]
fprintf('Distillate: %g\n\n', D)
B = F - D;                        %Calculate bottoms rate [mol/h]

%% Feed quality

y_fl =  q ./ (q-1) .* x1 - z1 ./ (q-1); %Feed line

%% Upper operating line

%Calculate RR Min

[x_inter, y_inter] = polyxpoly(x1, y1, x1, y_fl); %Find intersection point
m = (x_D - y_inter)/(x_D - x_inter);              %slope of operating line
b = x_D - m * x_D;                              %y-intercept upper operating line
RR_min = (x_D./b) - 1;              %RR min from the intercept of operating line
fprintf('RR_min value: %g\n' , RR_min);
%Calculate actual operating line
RR = R_param * RR_min;         %Real reflux ration
fprintf('reflux ratio: %g\n\n', RR)
y_UOL = ( RR/(RR+1) ).*x1 + (1/(1+RR))*x_D;  %Operating line
L_down = D * RR;               %Condensed liquid in condenser

%% Lower operating line

[x_inter2, y_inter2] = polyxpoly(x1, y_UOL, x1, y_fl); %Find intersection point
m1 = (x_B - y_inter2)/(x_B - x_inter2); %slope of operating line
b1 = x_B - m1 * x_B;                    %y-intercept upper operating line
V_b = 1 / (m1 - 1);                      %Calculate boilup ratio
V_up = V_b * B;                         %Vapour reboiled

%% Step function
```

```
y_UOL_func = @(x) ( RR/(RR+1) ).*x + (1/(1+RR))*x_D; %Upper operating line
y_LOL_func = @(x) m1 .* x + b1;                   %Lower operating line

i=1;
x_low(1) = x_B;
y_low(1) = x_B;

while (x_low(i) < x_inter2),
    y_low(i+1) = raoult(x_low(i),P);
    x_low(i+1) = fzero(@(x) y_low(i+1) - y_LOL_func(x), 0.5);
    i = i+1;
end

j=1;
y_up(1) = y_low(i);
x_up(1) = x_low(i);


while (x_up(j) < x_D),
    x_up(j+1) = fzero(@(x) y_up(j) - y_UOL_func(x), 0.7);
    y_up(j+1) = raoult(x_up(j+1),P);
    j = j+1;
end

N_theory = i + j - 1; %Theoretical tray number
eps = 0.7;            %Tray efficiency
N_stages = N_theory/eps; %Actual number of trays
fprintf('Number of stages: %g\n\n', N_stages)
feed_stage = i/N_theory * N_stages  %Calculate the feed stage

%% Plot for Raoult

figure(1);
plot([z1, x_inter2],[z1, y_inter2]); %Plot feed line
hold on;
plot(x1,y1);  %Plot equilibrium line
plot([x_D, x_inter2], [x_D, y_inter2]); %Plot upper operating line
plot([x_B, x_inter2], [x_B, y_inter2]); %Plot lower operating line
plot(x1,x1);  %Plot Diagonal
for k = 2:i
    plot([x_low(k-1) x_low(k-1)], [y_low(k-1) y_low(k)],'k');
    plot([x_low(k-1) x_low(k)], [y_low(k) y_low(k)],'k');

end
for l= 2:j
    plot([x_up(l-1) x_up(l)], [y_up(l-1) y_up(l-1)],'k');
    plot([x_up(l) x_up(l)], [y_up(l-1) y_up(l)],'k');
end
hold off;
xlim([0 1]); ylim([0 1]);
title('Mccabe-Thiele method with Raoults Law VLE');
xlabel('Mole fraction of HCN in liquid phase, x_{HCN}');
ylabel('Mole fraction of HCN in vapour phase, y_{HCN}');
legend('Feed line', 'Equilibrium line', 'Upper operating line', 'Lower operating line','location','southeast');

%% Principal dimensions of column

Height = 1.2 * 0.6 * N_stages; % Height of column
fprintf('Height distillation tower: %g\n\n', Height)

%Diameter of column.. Vmax will most likely be above the feed
%Find the density of the vapour at this point

V_above_feed = V_up + (1-q)*F; %[mol/h]
R = 8.134;                     %Gas constant
T_av = 80;         %Temperature at the middle of the column [C]
molar_dens = 133.32 * P / ( R * (T_av + 273.15)); %[mol/m^3]
V_vol = V_above_feed / (3600 * molar_dens); %[m^3/s]

%Assumption, 1 bar pressure and vapour is mostly steam
dens_steam = 0.590;                    %[kg/m^3]
dens_steam_unit = dens_steam * 2.0246 / 3.2808^3; % [lb/ft^3}
V_max_unit = 1 / sqrt(dens_steam_unit); %velocity of vapour [ft/s]
```

```
V_max = V_max_unit / 3.2808;            %vapour veloctiy [m/s]
Area = V_vol / V_max;                   %area of column
Diam = 2 * sqrt(Area/pi);               %diameter of column
fprintf('Diameter distillation tower: %g\n\n', Diam)


%% Energy balance

% properties
cpl1 = 71; % heat cap liquid 1 [J/mol K]
cpl2 = 75.3; % heat cap liquid 2  [J/mol K]
hvap1 = 28.1e3; % heat cap vap 1 [J/mol]
hvap2 = 40.65e3; % heat cap vap 1 [J/mol]

Ereboiler = V_up * hvap2;   %[J/h]
Econdenser = D * hvap1;     %[J/h]

[~, T_b]= raoult(z1,P);  % better code?!
 fprintf('Boiling point: %g\n\n', T_b)

% deltaT = abs(dummy_temp - Tin);
% Efeed = z1 * F * deltaT * cpl1 + (1-z1) * F * deltaT * cpl2; %[J/h]
%
% Tsteam = 151;
% diffT1 = abs( (dummy_temp + Tin) / 2 - Tsteam);
% reboiler1area = Efeed / (3600*570*diffT1);
reboiler2area = Ereboiler / (3600*570*51)
condenserarea = Econdenser / (3600*850*36)

%Calculate how much steam
deltaH_steam = 2107420; %Enthalpy of vapourisation of steam at 5 bar [J/kg]
mass_steam_hour = (Ereboiler)/deltaH_steam;
mass_steam_year = mass_steam_hour * 8000;

%Calculate how much brine needed for condenser
cp_brine_15 = 3500;                             %Heat capacity of 15% brine [J/kgK]
mflow_brine_hour = Econdenser / (cp_brine_15 * 16); %Mass flow of steam [kg/h]
mflow_brine_year = mflow_brine_hour * 8000;   %Mass steam for a year [kg]

%% Costs

%Capex

%Calculate cost of column
cost_column = 80320 * (Height ^ 0.76) * (Diam ^ 1.21);

%Cost for height exchangers
%cost_reboiler1 = 25000 * reboiler1area ^ 0.65;
cost_reboiler2 = 25000 * reboiler2area ^ 0.65;
cost_condenser = 25000 * condenserarea ^ 0.65;

%Total capex
capex = cost_column + cost_reboiler2 + cost_condenser;
%capex = cost_column + cost_reboiler1 + cost_reboiler2 + cost_condenser;
fprintf('capex cost distillation: %g\n\n', capex)

%Opex

%Steam
steam_price = 25/1000; %price of steam [$/kg]
cost_steam = mass_steam_year * steam_price; %Total cost of steam /year
cost_steam_10year = cost_steam * 10; %Total cost of steam for 10 years

%Brine
brine_price = 0.2/1000; %price of chilled water [$/kg];
cost_brine_year = mflow_brine_year * brine_price; %cost of brine per year
cost_brine_10year = cost_brine_year * 10; %total cost over 10 years

%Total Opex
opex = cost_steam_10year + cost_brine_10year;
fprintf('opex cost distillation: %g\n\n', opex)
```

```
%Total costs over 10 years
totex = capex + opex;
fprintf('Total cost distillation: %g\n\n', totex)
end
function [y1, Tmix] = raoult(x1,P)
% calcultae x,y of the first component
% P in mmHg !!!!

param;

x2 = 1-x1;

AK1 = 1; %aktivitaetscoeff
AK2 = 1;

ps1 =@(t)10.^(a1-b1./(c1+t)); %antoines eq fuer ps in mmHg
ps2 =@(t)10.^(a2-b2./(c2+t));

Tmix = fzero(@(t)P-AK1.*ps1(t).*x1-AK2.*ps2(t).*x2,30);

satpres1 = ps1(Tmix);
satpres2 = ps2(Tmix);
y1 = (AK1 .* x1.*satpres1)./P;
y2 = (AK2 .* x2.*satpres2)./P;
end
```

Listing 7: File used in section 3.4 to model the distillation column with raoult approach.

```
function [totex, capex, opex, D, N_stages, Height, Diam, B, RR, RR_min] = mccabe_vanlaarfunc(z1, P_atm, q,
    R_param, x_D, x_B, F)
%% Distillation column
% This function calculates the dimensions of a distillation column using
% the Mccabe-Thiele method based on the Van Laar model for VLE
% The input paramaters are as follows:
% P = Pressure [mmHG]
% Tin = Temperature at which feed is delivered [C]
% q = feed quality
% z1 = molar fraction of HCN in feed
% R_param = multiplying factor of reflx ratio

% x_D = Fraction of HCN in distillate
% x_B = fraction of HCN in bottoms
% F = feed rate [mol/h];

%Convert pressure from atmospheres to mmHg
P = P_atm * 760;

x1 = linspace(0.0000001,1,1000);  %Calculate the equilibrium line using vL VLE
y1 = zeros(0,length(x1));
Tmix = zeros(0, length(x1));

for g = 1:length(x1)
[y1(g), Tmix(g)] = vL(x1(g),P);
end

%% Mass Balances

D = F * (z1 - x_B) / (x_D - x_B); %Calculate distillate rate [mol/h]
fprintf('Distillate: %g\n\n', D)
B = F - D;                        %Calculate bottoms rate [mol/h]

%% Feed line

y_fl =  q ./ (q-1) .* x1 - z1 ./ (q-1);  %Feed line

%% Upper operating line

%Calculate RR Min

[x_inter, y_inter] = polyxpoly(x1, y1, x1, y_fl); %Find intersection point
m = (x_D - y_inter)/(x_D - x_inter); %slope of operating line
b = x_D - m * x_D;      %y-intercept upper operating line
```

```
RR_min = (x_D/b) - 1;   %RR min from the intercept of operating line

%Calculate actual operating line
RR = R_param * RR_min;  %Theoretical reflux ratio
fprintf('reflux ratio: %g\n\n', RR)
y_UOL = (RR/(RR+1) ).*x1 + (1/(1+RR))*x_D;  %Upper operating line

%% Lower operating line

[x_inter2, y_inter2] = polyxpoly(x1, y_UOL, x1, y_fl); %Find intersection point
m1 = (x_B - y_inter2)/(x_B - x_inter2); %slope of operating line
b1 = x_B - m1 * x_B;                    %y-intercept upper operating line
V_b = 1 / (m1 - 1);                     %Boilup ratio
V_up = V_b * B;                         %Vapour reboiled

%% Step function

y_UOL_func = @(x) ( RR/(RR+1) ).*x + (1/(1+RR))*x_D; %Upper operating line
y_LOL_func = @(x) m1 .* x + b1;                   %Lower operating line

i=1;
x_low(1) = x_B;
y_low(1) = x_B;

while (y_low(i) < y_inter2),
    y_low(i+1) = vL(x_low(i),P);
    x_low(i+1) = fzero(@(x) y_low(i+1) - y_LOL_func(x), 0.5);
    i = i+1;
end

j=1;
y_up(1) = y_low(i);
x_up(1) = x_low(i);

while (x_up(j) < x_D),
    x_up(j+1) = fzero(@(x) y_up(j) - y_UOL_func(x), 0.7);
    y_up(j+1) = vL(x_up(j+1),P);
    j = j+1;
end

N_theory = i + j - 1;      %Theoretical tray number
eps = 0.7;                 %Tray efficiency
N_stages = N_theory/eps;   %Actual number of trays
fprintf('Number of stages: %g\n\n', N_stages)
feed_stage = i/N_theory * N_stages  %Calculate the feed stage

%% Plot for van Laar

figure(2);
plot([z1, x_inter2],[z1, y_inter2]);    %Plot feed line
hold on;
plot(x1,y1);                            %Plot equilibrium line
plot([x_D, x_inter2], [x_D, y_inter2]); %Plot upper operating line
plot([x_B, x_inter2], [x_B, y_inter2]); %Plot lower operating line
plot(x1,x1);                            %Plot Diagonal
for k = 2:i
    plot([x_low(k-1) x_low(k-1)], [y_low(k-1) y_low(k)],'k');
    plot([x_low(k-1) x_low(k)], [y_low(k) y_low(k)],'k');

end
for l= 2:j
    plot([x_up(l-1) x_up(l)], [y_up(l-1) y_up(l-1)],'k');
    plot([x_up(l) x_up(l)], [y_up(l-1) y_up(l)],'k');
end
xlim([0 1]); ylim([0 1]);
hold off;
title('Mccabe-Thiele method with Van Laar VLE');
xlabel('Mole fraction of HCN in liquid phase, x_{HCN}');
ylabel('Mole fraction of HCN in vapour phase, y_{HCN}');
legend('Feed line', 'Equilibrium line', 'Upper operating line', 'Lower operating line','location','southeast');

%% Principal dimensions of column
```

```
Height = 1.2 * 0.6 * N_stages; % Height of column
fprintf('Height distillation tower: %g\n\n', Height)


%Diameter of column.. Vmax will most likely be above the feed
%Find the density of the vapour at this point

V_above_feed = V_up + (1-q)*F; %[mol/h]
R = 8.134; %Gas constant
T_feed = 80;
molar_dens = 133.32 * P / ( R * (T_feed + 273.15)); %[mol/m^3]
V_vol = V_above_feed / (3600 * molar_dens); %[m^3/s]

%Assumption, 1 bar pressure and vapour is mostly steam
dens_steam = 0.590;                                %[kg/m^3]
dens_steam_unit = dens_steam * 2.0246 / 3.2808^3; %[lb/ft^3}
V_max_unit = 1 / sqrt(dens_steam_unit);           %[ft/s]
V_max = V_max_unit / 3.2808;                       %[m/s]
Area = V_vol / V_max;                              %[m^2]
Diam = 2 * sqrt(Area/pi);                          %Diameter [m]
fprintf('Diameter distillation tower: %g\n\n', Diam)


%% Energy balance

% properties
cpl1 = 71; % heat cap liquid 1 [J/mol K]
cpl2 = 75.3; % heat cap liquid 2  [J/mol K]
hvap1 = 28.1e3; % heat cap vap 1 [J/mol]
hvap2 = 40.65e3; % heat cap vap 1 [J/mol]

Ereboiler = V_up * hvap2;           %[J/h]
Econdenser = D * hvap1; %[J/h]

[~, T_b]= vL(z1,P);
fprintf('Boiling point: %g\n\n', T_b)

% deltaT = abs(dummy4-Tin);
% Efeed = z1*F*deltaT*cpl1+(1-z1)*F*deltaT*cpl2; %[J/h]
%
% Tsteam = 151;
% diffT1 = abs((dummy4+Tin)/2-Tsteam);
% reboiler1area = Efeed/3600/570/diffT1;
reboiler2area = Ereboiler/3600/570/51
condenserarea = Econdenser/3600/850/36

%Calculate how much steam
deltaH_steam = 2107420; %Enthalpy of vapourisation of steam at 5 bar [J/kg]
mass_steam_hour = (Ereboiler)/deltaH_steam;
mass_steam_year = mass_steam_hour * 8000;

%Calculate how much brine needed for condenser
cp_brine_15 = 3500; %Heat capacity of 15% brine [J/kgK]
mflow_brine_hour = Econdenser / (cp_brine_15 * 16); %Mass flow of steam [kg/h]
mflow_brine_year = mflow_brine_hour * 8000;  %Mass steam for a year [kg]

%% Costs

%Capex

%Calculate cost of column
cost_column = 80320 * (Height ^ 0.76) * (Diam ^ 1.21);

%Cost for height exchangers
%cost_reboiler1 = 25000 * reboiler1area ^ 0.65;
cost_reboiler2 = 25000 * reboiler2area ^ 0.65;
cost_condenser = 25000 * condenserarea ^ 0.65;

%Total capex
capex = cost_column + cost_reboiler2 + cost_condenser;
fprintf('capex cost distillation: %g\n\n', capex)
%Opex
```

```
%Steam
steam_price = 25/1000; %price of steam [$/kg]
cost_steam = mass_steam_year * steam_price; %Total cost of steam /year
cost_steam_10year = cost_steam * 10; %Total cost of steam for 10 years

%Brine
brine_price = 0.2/1000; %price of chilled water [$/kg];
cost_brine_year = mflow_brine_year * brine_price; %cost of brine per year
cost_brine_10year = cost_brine_year * 10; %total cost over 10 years

%Total Opex
opex = cost_steam_10year + cost_brine_10year;
fprintf('opex cost distillation: %g\n\n', opex)

%Total costs over 10 years
totex = capex + opex; fprintf('Total cost distillation: %g\n\n', totex)
end
function [y1, Tmix] = vL(x1,P)
% calcultae x,y of the first component
% P in mmHg !!!!

param;

x2 = 1-x1;

AK1 = exp(A1*((A2*x2)./(A1*x1+A2*x2))^2); %aktivitaetscoeff
AK2 = exp(A2*((A1*x1)./(A1*x1+A2*x2))^2);

ps1 =@(t)10.^(a1-b1./(c1+t)); %antoines eq fuer ps in mmHg
ps2 =@(t)10.^(a2-b2./(c2+t));

Tmix = fzero(@(t)P-AK1.*ps1(t).*x1-AK2.*ps2(t).*x2,30);

satpres1 = ps1(Tmix);
satpres2 = ps2(Tmix);
y1 = (AK1 .* x1 .* satpres1)./P;
y2 = (AK2 .* x2 .* satpres2)./P;

end
```

Listing 8: File used in section 3.4 to model the distillation column with van laar approach.

```
function optimisation

%B_vL(1)    ...flow rate
%B_vL(2)    ...total cost

guess = 300000; %[mol/h]
options = optimset('display', 'off');
wash_water = lsqnonlin(@(B_vL) optim(B_vL), guess, [], [], options);

end
function [cost_tot] = optim(B_vL)

R_param = 1.1;
q = 0.8;
P_atm = 1;
x_D = 0.995;
x_B = 10/1000000;

[L_output, y_HCN_abs2_out, cost_abs] = absorber2(B_vL);

L_output = L_output * 3600; %next file in mol/h

[totex_vanlaar, ~, ~, ~, ~, ~, B_vL, ~, ~] = mccabe_vanlaarfunc(y_HCN_abs2_out, P_atm, q, R_param, x_D, x_B,
      L_output);

cost_tot = abs(totex_vanlaar + cost_abs(1));
fprintf('flow rate B_vL: %g\n', B_vL)

end
```

Listing 9: File used in section 3.4 to optimise distillation and HCN absorber.

```
%% Properties HCN
M_HCN = 27.03; % [g/mol]
p_crit_HCN = 5390e3; % critical pressure [Pa]
T_crit_HCN = 456.65; % critical Temperature [K]
ac_fac_HCN = 0.4001; % acentric factor HCN [-]
a1 = 7.52823; % antoines coefficients for Celsius and mmHG
b1= 1329.49;
c1 = 260.418;

%% Properties H2O
M_H2O = 18; % [g/mol]
p_crit_H2O = 22119.2e3; % critical pressure water [Pa]
T_crit_H2O = 647.3; % critical Temperature water [K]
ac_fac_H2O = 0.3644; % acentric factor water [-]
a2 = 8.07131; % antoines coeff beide fuer Celsius
b2 = 1730.630;
c2= 233.426;

%% phys properties

A1 = 2.0536;
A2 = 1.6800;  % van laar coeff japanese
```

Listing 10: File used in section 3.4 for the parameters and physical properties.